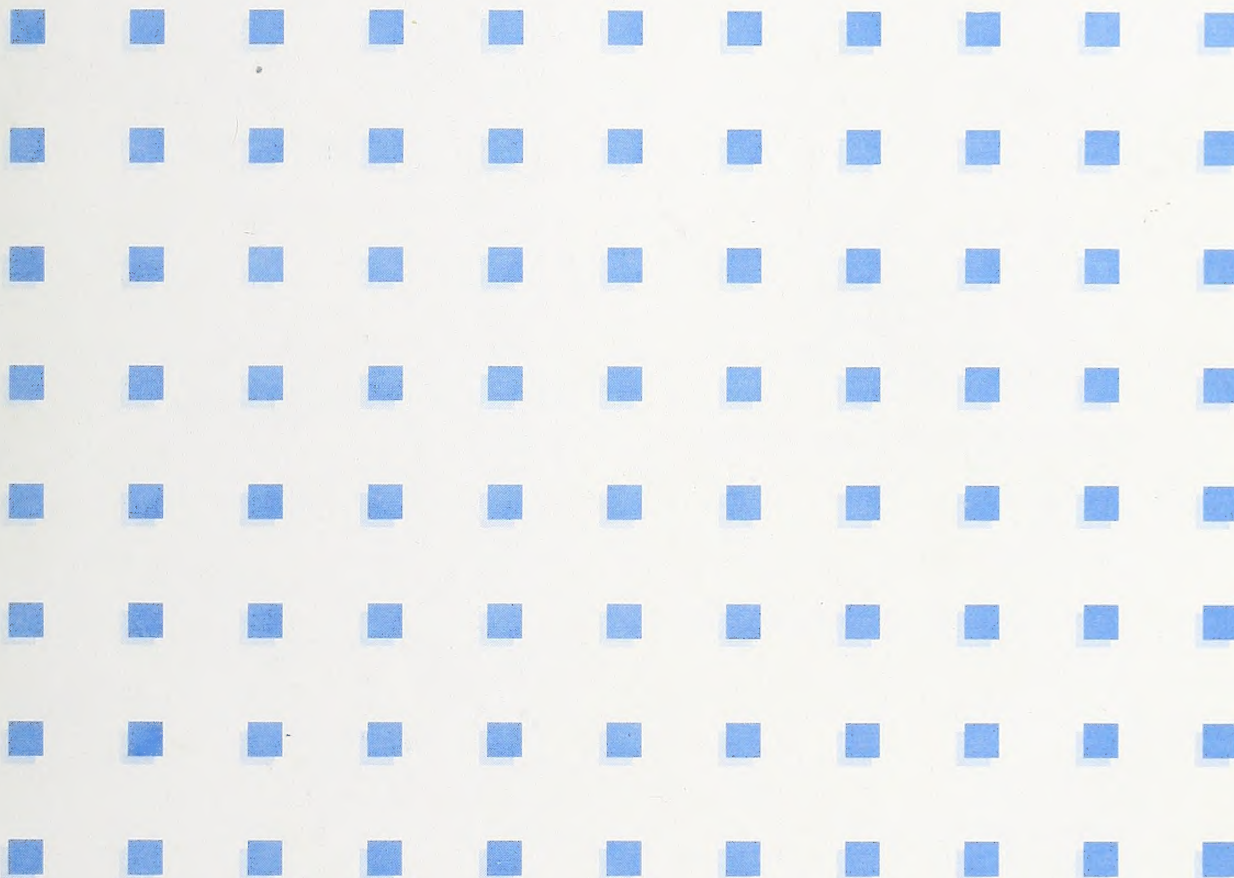


Information Technology:

The Fifth Text REtrieval Conference (TREC-5)

E. M. Voorhees and
D. K. Harman, Editors



U.S. Department of Commerce
Technology Administration
National Institute of Standards and Technology

NIST

QC
100
.U57
NO. 500-238
1997

The National Institute of Standards and Technology was established in 1988 by Congress to “assist industry in the development of technology . . . needed to improve product quality, to modernize manufacturing processes, to ensure product reliability . . . and to facilitate rapid commercialization . . . of products based on new scientific discoveries.”

NIST, originally founded as the National Bureau of Standards in 1901, works to strengthen U.S. industry's competitiveness; advance science and engineering; and improve public health, safety, and the environment. One of the agency's basic functions is to develop, maintain, and retain custody of the national standards of measurement, and provide the means and methods for comparing standards used in science, engineering, manufacturing, commerce, industry, and education with the standards adopted or recognized by the Federal Government.

As an agency of the U.S. Commerce Department's Technology Administration, NIST conducts basic and applied research in the physical sciences and engineering, and develops measurement techniques, test methods, standards, and related services. The Institute does generic and precompetitive work on new and advanced technologies. NIST's research facilities are located at Gaithersburg, MD 20899, and at Boulder, CO 80303. Major technical operating units and their principal activities are listed below. For more information contact the Publications and Program Inquiries Desk, 301-975-3058.

Office of the Director

- National Quality Program
- International and Academic Affairs

Technology Services

- Standards Services
- Technology Partnerships
- Measurement Services
- Technology Innovation
- Information Services

Advanced Technology Program

- Economic Assessment
- Information Technology and Applications
- Chemical and Biomedical Technology
- Materials and Manufacturing Technology
- Electronics and Photonics Technology

Manufacturing Extension Partnership Program

- Regional Programs
- National Programs
- Program Development

Electronics and Electrical Engineering Laboratory

- Microelectronics
- Law Enforcement Standards
- Electricity
- Semiconductor Electronics
- Electromagnetic Fields¹
- Electromagnetic Technology¹
- Optoelectronics¹

Chemical Science and Technology Laboratory

- Biotechnology
- Physical and Chemical Properties²
- Analytical Chemistry
- Process Measurements
- Surface and Microanalysis Science

Physics Laboratory

- Electron and Optical Physics
- Atomic Physics
- Optical Technology
- Ionizing Radiation
- Time and Frequency¹
- Quantum Physics¹

Materials Science and Engineering Laboratory

- Intelligent Processing of Materials
- Ceramics
- Materials Reliability¹
- Polymers
- Metallurgy
- NIST Center for Neutron Research

Manufacturing Engineering Laboratory

- Precision Engineering
- Automated Production Technology
- Intelligent Systems
- Fabrication Technology
- Manufacturing Systems Integration

Building and Fire Research Laboratory

- Structures
- Building Materials
- Building Environment
- Fire Safety Engineering
- Fire Science

Information Technology Laboratory

- Mathematical and Computational Sciences²
- Advanced Network Technologies
- Computer Security
- Information Access and User Interfaces
- High Performance Systems and Services
- Distributed Computing and Information Services
- Software Diagnostics and Conformance Testing

¹ At Boulder, CO 80303.

² Some elements at Boulder, CO.

Information Technology:

The Fifth Text REtrieval Conference (TREC-5)

E. M. Voorhees and
D. K. Harman, Editors

Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-0001

November 1997



U.S. Department of Commerce

William M. Daley, Secretary

Technology Administration

Gary R. Bachula, Acting Under Secretary for Technology

National Institute of Standards and Technology

Robert E. Hebner, Acting Director

Reports on Information Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) stimulates U.S. economic growth and industrial competitiveness through technical leadership and collaborative research in critical infrastructure technology, including tests, test methods, reference data, and forward-looking standards, to advance the development and productive use of information technology. To overcome barriers to usability, scalability, interoperability, and security in information systems and networks, ITL programs focus on a broad range of networking, security, and advanced information technologies, as well as the mathematical, statistical and computational sciences. This Special Publication 500 series reports on ITL's research in tests and test methods for information technology, and its collaborative activities with industry, government, and academic organizations.

National Institute of Standards and Technology
Special Publication 500-238
Natl. Inst. Stand. Technol.
Spec. Publ. 500-238
769 pages (Nov. 1997)
CODEN: NSPUE2

U.S. Government Printing Office
Washington: 1997

For sale by the Superintendent of Documents
U.S. Government Printing Office, Washington, DC 20402-9325

Foreword

This report constitutes the proceedings of the fifth Text REtrieval Conference (TREC-5) held in Gaithersburg, Maryland, November 20–22, 1996. The conference was co-sponsored by the National Institute of Standards and Technology (NIST) and the Defense Advanced Research Projects Agency (DARPA), and was attended by 145 people. Thirty-eight groups including participants from nine different countries and ten companies were represented. The conference was the fifth in an on-going series of workshops to evaluate new technologies in text retrieval.

The workshop included plenary sessions, discussion groups, a poster session, and demonstrations. Because the participants in the workshop drew on their personal experiences, they sometimes cited specific vendors and commercial products. The inclusion or omission of a particular company or product implies neither endorsement nor criticism by NIST.

The sponsorship of the Information Technology Office of the Defense Advanced Research Projects Agency is gratefully acknowledged, as is the tremendous work of the program committee.

Ellen Voorhees,
Donna Harman
October 29, 1997

TREC-5 Program Committee

Donna Harman, NIST, chair
Nick Belkin, Rutgers University
Chris Buckley, Cornell University
Jamie Callan, University of Massachusetts at Amherst
Susan Dumais, Bellcore
Darryl Howard, U.S. Department of Defense
David Lewis, AT&T Research
John Prange, U.S. Department of Defense
Steve Robertson, City University, UK
Alan Smeaton, Dublin City University, Ireland
Karen Sparck Jones, University of Cambridge, UK
Richard Tong, Sageware, Inc.
Howard Turtle, West Publishing
Ellen Voorhees, NIST
Ross Wilkinson, Royal Melbourne Institute of Technology

TABLE OF CONTENTS

ABSTRACT	ix
-----------------------	-----------

PAPERS

1. Overview of the Fifth Text REtrieval Conference (TREC-5)	1
E. Voorhees, D. Harman (National Institute of Standards and Technology)	
2. TREC-5 Interactive Track Report	29
P. Over (National Institute of Standards and Technology)	
3. Spanish and Chinese Document Retrieval in TREC-5	57
A. Smeaton (Dublin City University), R. Wilkinson (Royal Melbourne Institute of Technology)	
4. Report on the TREC-5 Confusion Track	65
P. Kantor (Rutgers University), E. Voorhees (National Institute of Standards and Technology)	
5. The TREC-5 Filtering Track	75
D. Lewis (AT&T Labs--Research)	
6. NLP Track at TREC-5	97
T. Strzalkowski (GE Corporate Research & Development)	
K. Sparck Jones (University of Cambridge)	
7. The TREC-5 Database Merging Track	103
E. Voorhees (National Institute of Standards and Technology)	
8. Using Query Zoning and Correlation Within SMART: TREC-5	105
C. Buckley, A. Singhal, M. Mitra (Cornell University)	
9. INQUERY at TREC-5	119
J. Allan, J. Callan, B. Croft, L. Ballesteros, J. Broglio, J. Xu, H. Shu (University of Massachusetts, Amherst)	
10. TREC-5 English and Chinese Retrieval Experiments using PIRCS	133
K.L. Kwok, L. Grunfeld (Queens College, CUNY)	
11. Okapi at TREC-5	143
M. Beaulieu, M. Gatford, X. Huang, S. Robertson, S. Walker, P. Williams (City University, London)	
12. Xerox TREC-5 Site Report: Routing, Filtering, NLP, and Spanish Tracks	167
D. Hull, G. Grefenstette, B. Schulze, E. Gaussier (Rank Xerox Research Center), H. Schütze (Xerox PARC), J. Pedersen (Verity Corporation)	

13. Term importance, Boolean conjunct training, negative terms, and foreign language retrieval: probabilistic algorithms at TREC-5	181
F. Gey, A. Chen, J. He, L. Xu, J. Meggs (University of California, Berkeley)	
14. Berkeley Chinese Information Retrieval at TREC-5: Technical Report	191
J. He, J. Xu, A. Chen, J. Meggs, F. Gey (University of California, Berkeley)	
15. TREC-5 Experiments at Dublin City University: Query Space Reduction, Spanish & Character Shape Encoding	197
F. Kellely, A. Smeaton (Dublin City University)	
16. The MDS Experiments for TREC5	209
M. Kaszkiel, P. Vines, R. Wilkinson, J. Zobel (RMIT)	
17. SPIDER Retrieval System at TREC-5	217
J. Ballerini, M. Büchel, R. Domenig, D. Knaus, B. Mateev, E. Mittendorf, P. Schäuble, P. Sheridan, M. Wechsler (Swiss Federal Institute of Technology (ETH))	
18. Ad Hoc Experiments Using EUREKA	229
A. Lu, M. Ayoub, J. Dong (Lexis-Nexis)	
19. InTEXT Automatic Query Enhancement in TREC-5	241
R. Jones, M. Burnett, L. Pape (InTEXT Systems)	
20. Experiments on Routing, Filtering and Chinese Text Retrieval in TREC-5	247
C. Ngo, K. Lai (Information Technology Institute, Singapore)	
21. Rutgers Interactive Track at TREC-5	257
N. Belkin, A. Cabezas, C. Cool, K. Kim, K. Ng, S. Park, R. Pressman, S. Rieh, P. Savage, H. Xie (Rutgers University)	
22. Interactive Substring Retrieval (MultiText Experiments for TREC-5)	267
C. Clarke, G. Cormack (University of Waterloo)	
23. V-Twin: A Lightweight Engine for Interactive Use	279
D. Rose, C. Stevens (Apple Computer, Inc.)	
24. Natural Language Information Retrieval: TREC-5 Report	291
T. Strzalkowski, F. Lin, J. Wang (GE Corporate Research and Development), L. Guthrie, J. Leistensnider, J. Wilding (Lockheed Martin Corporation), J. Karlgren, T. Straszheim (New York University), J. Perez-Carballo (Rutgers University)	
25. CLARIT Compound Queries and Constraint-Controlled Feedback in TREC-5 Ad-Hoc Experiments	315
N. Milic-Frayling, D. Evans (CLARITECH Corporation) X. Tong, C. Zhai (Carnegie Mellon University)	

26. Experiments on Chinese Text Indexing--CLARIT TREC-5 Chinese Track Report	335
X. Tong, C. Zhai (Carnegie Mellon University)	
N. Milić-Frayling, D. Evans (CLARITECH Corporation)	
27. OCR Correction and Query Expansion for Retrieval on OCR Data--CLARIT TREC-5 Confusion Track Report	341
X. Tong, C. Zhai (Carnegie Mellon University)	
N. Milić-Frayling, D. Evans (CLARITECH Corporation)	
28. Evaluation of Syntactic Phrase Indexing--CLARIT NLP Track Report	347
C. Zhai, X. Tong (Carnegie Mellon University)	
N. Milić-Frayling, D. Evans (CLARITECH Corporation)	
29. ANU/ACSys TREC-5 Experiments	359
D. Hawking, P. Thistlewaite, P. Bailey (Australian National University)	
30. Parallel Techniques For Efficient Searching Over Very Large Text Collections	377
B. Mamalis, P. Spirakis, B. Tampakas (Computer Technology Institute)	
31. Document Retrieval Using The MPS Information Server (A Report on the TREC-5 Experiment)	391
F. Schiettecatte (FS Consulting, Inc.)	
32. Using Relevance Feedback within the Relational Model for TREC-5	405
D. Grossman (Office of Information Technology), J. Reichart (P2000Technology Inc.),	
A. Chowdhury, C. Lundquist (George Mason University), D. Holmes (NCR),	
O. Frieder (Florida Institute of Technology)	
33. TREC-5 Ad Hoc Retrieval Using K Nearest-Neighbors Re-Scoring	415
E. Chan, S. Garcia, S. Roukos (IBM T.J. Watson Research Center)	
34. The GURU System in TREC-5	427
Y. Ravin (IBM T.J. Watson Research Center)	
35. Mercure02: adhoc and routing tasks	429
M. Boughanem (Université de Limoges), C. Soule-Dupuy (IRIT Toulouse)	
36. Information Retrieval and Trainable Natural Language Processing	433
J. Burger, J. Aberdeen, D. Palmer (The MITRE Corporation)	
37. Using Bayesian Networks as Retrieval Engines	437
M. Indrawan, D. Ghazfan, B. Srinivasan (Monash University, Australia)	
38. CRL English Routing System for TREC-5	445
J. Cowie, Z. Guan (New Mexico State University)	
39. New Experiments In Cross-Language Text Retrieval At NMSU's Computing Research Lab	447
M. Davis (New Mexico State University)	

40. Experiments with TREC using the Open Text Livelink Engine	455
L. Fitzpatrick, M. Dent, G. Promhouse (Open Text Corporation)	
41. Data Fusion of Machine-Learning Methods for the TREC5 Routing Task (and other work)	477
K. Ng, H. Hirsh, P. Kantor (Rutgers University), D. Loewenstern (Bell Laboratories), C. Basu (Bellcore)	
42. Report on the TREC-5 Experiment: Data Fusion and Collection Fusion	489
J. Savoy, A. Le Calvé, D. Vrajitoru (Université de Neuchâtel, Switzerland)	
43. Using Relevance to Train a Linear Mixture of Experts	503
C. Vogt, G. Cottrell, R. Belew, B. Bartell (University of California, San Diego)	
44. Report on the Glasgow IR group (glair4) submission	517
M. Sanderson, I. Ruthven (University of Glasgow)	
45. Metric Multidimensional Information Space	521
G. Newby (University of Illinois)	
46. Corpus Analysis for TREC 5 Query Expansion	537
S. Gauch, J. Wang (University of Kansas)	
47. Alignment of Spanish and English TREC Topic Descriptions	547
D. Oard (University of Maryland)	
48. An Investigation of Relevance Feedback Using Adaptive Linear and Probabilistic Models	555
R. Sumner, Jr., W. Shaw, Jr. (University of North Carolina)	

APPENDICES

A. TREC-5 Results	A-1
B. Summary Performance Comparisons TREC-2, TREC-3, TREC-4, TREC-5	B-1
K. Sparck Jones (University of Cambridge)	

Abstract

This report constitutes the proceedings of the fifth Test REtrieval Conference (TREC-5) held in Gaithersburg, Maryland, November 20–22, 1996. The conference was co-sponsored by the National Institute of Standards and Technology (NIST) and the Defense Advanced Research Projects Agency (DARPA), and was attended by 145 people. Thirty-eight groups including participants from nine different countries and ten companies were represented.

The goal of the conference was to bring research groups together to discuss their work on a large test collection. The diversity of the participants meant that a wide variety of retrieval techniques were represented, including machine learning methods for query expansion and term weighting, sophisticated natural language processing techniques, and advanced pattern matching. Results were scored using a common evaluation package, so groups were able to compare the effectiveness of different techniques, and to discuss how differences between systems affected performance. In addition to the main evaluation, six additional evaluations, called “tracks”, allowed participants to focus on particular common subproblems.

The conference included paper sessions and discussion groups. This proceedings includes papers from most of the participants (some poster groups did not submit papers), track reports that define the problem addressed by the track plus summarize the main track results, and tables of individual group results. The TREC-5 proceedings web site also contains system descriptions that detail the timing and storage requirements of the different runs.

Overview of the Fifth Text REtrieval Conference (TREC-5)

Ellen M. Voorhees, Donna Harman
National Institute of Standards and Technology
Gaithersburg, MD 20899

1 Introduction

The fifth Text REtrieval Conference (TREC-5) was held at the National Institute of Standards and Technology (NIST) on November 20–22, 1996. The conference was co-sponsored by NIST and the Information Technology Office of the Defense Advanced Research Projects Agency (DARPA) as part of the TIPSTER Text Program.

TREC-5 is the latest in a series of workshops designed to foster research in text retrieval. For analyses of the results of previous workshops, see Sparck Jones [21], Tague-Sutcliffe and Blustein [23], and Harman [8]. In addition, the overview paper in each of the previous TREC proceedings summarizes the results of that TREC.

The TREC workshop series has the following goals:

- to encourage research in text retrieval based on large test collections;
- to increase communication among industry, academia, and government by creating an open forum for the exchange of research ideas;
- to speed the transfer of technology from research labs into commercial products by demonstrating substantial improvements in retrieval methodologies on real-world problems; and
- to increase the availability of appropriate evaluation techniques for use by industry and academia, including development of new evaluation techniques more applicable to current systems.

Table 1 lists the groups that participated in TREC-5. Thirty-eight groups including participants from nine different countries and ten companies were represented. The diversity of the participating groups has ensured that TREC represents many different approaches to text retrieval. The emphasis on individual experiments evaluated within a common setting has proven to be a major strength of TREC.

This paper serves as an introduction to the research described in detail in the remainder of the volume. The next section defines the common retrieval tasks performed in TREC-5. Sections 3 and 4 provide details regarding the test collections and the evaluation methodology used in TREC. Section 5 provides an overview of the retrieval results. The final section summarizes the main themes learned from the experiments.

2 The Tasks

Each of the TREC conferences has centered around two main tasks, the routing task and the ad hoc task. In addition, starting in TREC-4 a set of “tracks”, tasks that focus on particular subproblems of text retrieval, were introduced. TREC-5 continued the tracks started in TREC-4 and added a new track on natural language processing (NLP). This section describes the goals of the two main tasks in detail, and outlines the goals of each of the tracks. Readers are urged to consult the appropriate track report found later in these proceedings for details about individual tracks.

2.1 The routing task

The routing task in the TREC workshops investigates the performance of systems that use standing queries to search new streams of documents. These searches are similar to those required by news clipping services and library profiling systems. A true routing environment is simulated in TREC by using topics that have known relevant documents and testing on a completely new document set.

The training for the routing task is shown in the left-hand column of Figure 1. Participants are given a set of topics and a document set that includes known relevant documents for those topics. The topics consist of natural language text describing a user’s information need (see sec. 3.2 for details). The topics are used to create a set of queries (the actual input to

Table 1: Organizations participating in TREC-5

Apple Computer	MITRE
Australian National University	Monash University
CLARITECH Corporation	New Mexico State University (two groups)
City University	Open Text Corporation
Computer Technology Institute	Queens College, CUNY
Cornell University	Rank Xerox Research Center
Dublin City University	Rutgers University (two groups)
FS Consulting	Swiss Federal Institute of Technology (ETH)
GE/NYU/Rutgers/Lockheed Martin	Universite de Neuchatel
GSI-Erli	University of California, Berkeley
George Mason University	University of California, San Diego
IBM Corporation	University of Glasgow
IBM T.J. Watson Research Center	University of Illinois at Urbana-Champaign
Information Technology Institute, Singapore	University of Kansas
Institut de Recherche en Informatique de Toulouse	University of Maryland
Intext Systems	University of Massachusetts, Amherst
Lexis-Nexis	University of North Carolina
MDS at RMIT	University of Waterloo

the retrieval system) that are then used against the training documents. This is represented by Q1 in the diagram. Many Q1 query sets might be built to help adjust the retrieval system to the task, to create better weighting algorithms, and to otherwise prepare the system for testing. The result of the training is query set Q2, routing queries derived from the 50 routing topics (selected by NIST from the pool of training topics) and run against the test documents.

The testing phase of the routing task is shown in the middle column of Figure 1. The output of running Q2 against the test documents is the official test result for the routing task. In TREC-5, the routing topics were selected by choosing topics that had many relevant documents in the Associated Press (AP) collection and the test documents were articles extracted from the Foreign Broadcast Information Service (FBIS).

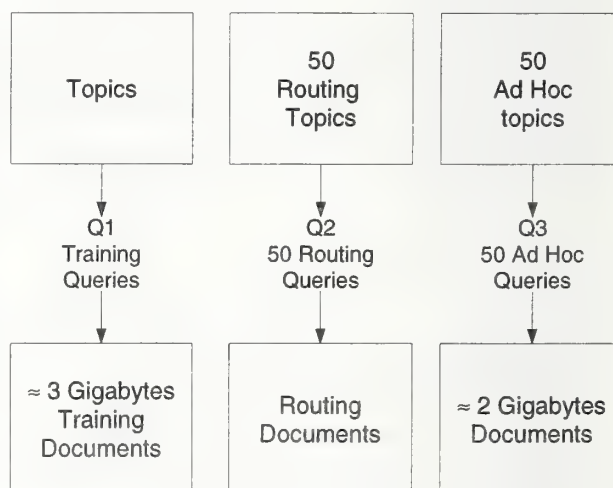


Figure 1: TREC main tasks.

2.2 The ad hoc task

The ad hoc task investigates the performance of systems that search a static set of documents using new topics. This task is similar to how a researcher might use a library — the collection is known but the questions likely to be asked are not known. The right-hand column of Figure 1 depicts how the ad hoc task is accomplished in TREC. Participants are given approximately two gigabytes worth of documents. They are also given 50 new topics. The set of relevant doc-

uments for these topics in the document set is not known at the time the participants receive the topics. Participants produce a new query set, Q3, from the ad hoc topics and run those queries against the ad hoc documents. The output from this run is the official test result for the ad hoc task. Topics 251–300 were created for the TREC-5 ad hoc task. The set of documents used in the task were those contained on Tipster Disk 2 and the new TREC Disk 4; see Section 3.1 for details about this document set.

2.3 Task guidelines

In addition to the task definitions, TREC participants are given a set of guidelines outlining acceptable methods of indexing, knowledge base construction, and generating queries from the supplied topics. In general, the guidelines are constructed to reflect an actual operational environment and to allow as fair as possible separation among the diverse query construction approaches. The allowable query construction methods in TREC-5 are divided into *automatic* methods, in which queries are derived completely automatically from the topic statements, and *manual* methods, which includes queries generated by all other methods. In contrast to previous TRECs, the definition of manual query construction methods in TREC-5 permitted users to look at individual documents retrieved by the ad hoc queries and then reformulate the queries based on the documents retrieved.¹

There are two levels of participation in TREC: category A, participation using the full dataset, or category B, participation using a reduced dataset (1/4 of the full document set). Groups could choose to do the routing task, the ad hoc task, or both, and were asked to submit the top 1000 documents retrieved for each topic for evaluation. Groups that performed the routing task were allowed to submit up to two official test results for judging. When two sets of results were sent, they could be made using different methods of creating queries, or different methods of searching with the same queries. Groups that performed the ad hoc task could submit up to two manual runs and up to two automatic runs. An additional constraint in this year's ad hoc task was that if any automatic results were submitted, at least one of the runs was required to use "short" topics (see sec. 3.2).

2.4 The tracks

One of the goals of TREC is to provide a common task evaluation that allows cross-system comparisons. This has proven to be a key strength in TREC. The second major strength is the loose definition of the two main tasks allowing a wide range of experiments.

¹Previous TRECs defined a third query construction method, *interactive*, for these types of runs. However, the interactive track in TREC-5 evolved such that these simple "manual feedback" runs did not fit well within the track's focus. The program committee redefined the manual query construction method to give the participants who were interested in studying manual feedback methods a home in TREC-5. Since both one-time and feedback runs are included in the single category of manual methods, care must be taken when comparing the results of manual runs.

The addition of secondary tasks (tracks) in TREC-4 combined these strengths by creating a common evaluation for tasks that are either related to the main tasks, or are a more focussed implementation of those tasks. Each of the tracks started in TREC-4 continued in TREC-5. In addition, a new track that focussed on using natural language processing techniques to improve retrieval performance was begun, and a "pre-track" laid the groundwork for the debut of the very large corpus (target of 20 GB of text) track in TREC-6.

TREC participants were free to turn in results for any, or all, or none, of the tracks. Each track had a set of guidelines developed under the direction of the track coordinator. The set of tracks and their primary goals are listed below. See the track reports elsewhere in this proceedings for a more complete description of each track.

Confusion: The confusion track investigates how retrieval performance is affected by noisy or "confused" data. In this running of the track, participants performed *known-item* searches; that is, they searched for particular previously identified documents in three versions of documents. The three versions of the documents were the original documents, the documents that resulted after the originals were subjected to an optical character recognition (OCR) process with a character error rate of approximately 5%, and the documents produced through OCR with a 20% error rate (caused by down-sampling the image before doing the OCR).

Database Merging: The database merging track investigates methods for producing a single document ranking for queries when the underlying data sets consists of separate document collections. The TREC-5 track had an explicit focus on accomplishing the distributed search *without* searching every document collection for every query. That is, part of the task was to define a method that selects some proper subset of the document collections to be searched for a given query.

Filtering: The filtering task is a routing task in which the system must decide whether or not to retrieve each individual document. Instead of producing a list of documents ranked according to the presumed similarity to a query, filtering systems retrieve an unordered set of documents for each query. The quality of the retrieved set is computed as a function of the benefit of a re-

trieved relevant document and the cost of a retrieved irrelevant document.

Interactive: The high-level goal of the interactive track is the investigation of searching as an interactive task by examining the process as well as the outcome.

Multilingual: The multilingual track investigates retrieval performance when the text (both documents and topics) is in a language other than English. The TREC-5 track contained both a Spanish task, which had also been run in TREC-4, and a Chinese task, which was introduced in TREC-5.

NLP: The NLP track was initiated to explore whether the natural language processing (NLP) techniques available today are mature enough to have an impact on IR, and specifically whether they can offer an advantage over purely quantitative retrieval methods.

3 The Test Collections

Like most traditional retrieval collections, there are three distinct parts to the collections used in TREC: the documents, the questions or topics, and the relevance judgments or "right answers." This section describes each of these pieces for the collections used in the TREC-5 main tasks.

3.1 Documents

TREC documents are distributed on CD-ROM's with approximately 1 GB of text on each, compressed to fit. For TREC-5, Disks 1, 2 and 3 were all available as training material (see Table 2) and Disk 2 and new Disk 4 were used for the ad hoc task. Additional new data (also shown in Table 2) was used for testing in the routing task.

Documents are tagged using SGML to allow easy parsing (see fig. 2). The documents in the different datasets have been tagged with identical major structures, but they have different minor structures. The philosophy in the formatting at NIST has been to preserve as much of the original structure as possible, while providing enough consistency to allow simple decoding of the data. Both as part of the philosophy of leaving the data as close to the original as possible, and because it is impossible to check all the data manually, many "errors" remain in the data. The error-checking done at NIST has concentrated on allowing readability of the data rather than on correcting content. This means that there have been automated

checks for control characters, special symbols, foreign language characters, for correct matching of the begin and end document tags, and for complete "DOCNO" fields (the field that gives the unique TREC identifier for the document). The types of "errors" remaining include fragment sentences, strange formatting around tables or other "non-textual" items, misspellings, etc.

The data on disk 4 and the FBIS routing test data are new TREC document sets. The *Federal Register* is the official record of the executive branch of the U.S. Government. Similarly, the *Congressional Record* is the proceedings of the legislative branch of the U.S. Government; the copy of the 103rd *Congressional Record* was obtained from Dean Wilder of the Library of Congress. The *Financial Times* articles were obtained from the Financial Times through the University of Glasgow. The Foreign Broadcast Information Service provides (English translations of) selected non-U.S. broadcast and print publications. The documents used in the routing test were mostly from the early 1990's and were provided for TREC use by the Foreign Broadcast Information Service.

3.2 Topics

In designing the TREC task, there was a conscious decision made to provide "user need" statements rather than more traditional queries. Two major issues were involved in this decision. First, there was a desire to allow a wide range of query construction methods by keeping the topic (the need statement) distinct from the query (the actual text submitted to the system). The second issue was the ability to increase the amount of information available about each topic, in particular to include with each topic a clear statement of what criteria make a document relevant.

The topics used in TREC-1 and TREC-2 (topics 1-150) were very detailed, containing multiple fields and lists of concepts related to the subject of the topics. The ad hoc topics used in TREC-3 (151-200) were not only much shorter, but also were missing the complex structure of the earlier topics. Nonetheless, participants in TREC-3 felt that the topics were still too long compared with what users normally submit to operational retrieval systems. Therefore the TREC-4 topics (201-250) were made even shorter: a single field consisting of a one sentence description of the information need. Figure 3 on page 7 gives a sample topic from each of these sets.

One of the conclusions reached in TREC-4 was that the much shorter topics caused both manual

Table 2: Document collection statistics. Words are strings of alphanumeric characters. No stop words were removed and no stemming was performed.

	Size (megabytes)	# Docs	Median # Terms/Doc	Mean # Terms/Doc
Disk 1				
<i>Wall Street Journal</i> , 1987–1989	267	98,732	245	434.0
<i>Associated Press</i> newswire, 1989	254	84,678	446	473.9
<i>Computer Selects</i> articles, Ziff-Davis	242	75,180	200	473.0
<i>Federal Register</i> , 1989	260	25,960	391	1315.9
abstracts of U.S. DOE publications	184	226,087	111	120.4
Disk 2				
<i>Wall Street Journal</i> , 1990–1992 (WSJ)	242	74,520	301	508.4
<i>Associated Press</i> newswire (1988) (AP)	237	79,919	438	468.7
<i>Computer Selects</i> articles, Ziff-Davis (ZIFF)	175	56,920	182	451.9
<i>Federal Register</i> (1988) (FR88)	209	19,860	396	1378.1
Disk 3				
<i>San Jose Mercury News</i> , 1991	287	90,257	379	453.0
<i>Associated Press</i> newswire, 1990	237	78,321	451	478.4
<i>Computer Selects</i> articles, Ziff-Davis	345	161,021	122	295.4
U.S. patents, 1993	243	6,711	4445	5391.0
Disk 4				
the <i>Financial Times</i> , 1991–1994 (FT)	564	210,158	316	412.7
<i>Federal Register</i> , 1994 (FR94)	395	55,630	588	644.7
<i>Congressional Record</i> , 1993 (CR)	235	27,922	288	1373.5
Routing Test Data				
Foreign Broadcast Information Service (FBIS)	470	130,471	322	543.6

and automatic systems trouble, and that there were issues associated with using short topics in TREC that needed further investigation [9]. Accordingly, the TREC-5 ad hoc topics re-introduced the title and narrative fields (see fig. 4, page 8), although, as shown in Table 3, the length of the topics as measured by number of words was generally shorter than in TREC-3.

Groups who performed automatic ad hoc runs were required to use a short version of the topics, just the “Description” field, for one of their runs. These runs are tagged as “short, automatic” runs in the results section; automatic runs that used the entire topic are tagged as “long, automatic” runs. Manual runs had no length requirements, and are assumed to be based on the entire topic text. The effect of the different lengths on retrieval performance is described in Section 5.

As was true for TREC-3 and TREC-4, each TREC-5 ad hoc topic was constructed by the same person who performed all relevance assessments for that topic (with a few exceptions). Assessors were asked to come to NIST with already-constructed top-

ics. The assessors used these topics to search (part of) the TREC-5 ad hoc collection (using NIST’s ZPRISE system) and to make an initial set of relevance assessments. NIST personnel used these assessments to select the final set of 50 topics from approximately 150 candidate topics, based mainly on how many relevant documents were found in the search. Candidate topics that retrieved too many or too few relevant documents were rejected. Candidate topics were also rejected if they seemed ambiguous. Once the final set of topics were selected, the initial relevance assessments were discarded.

3.3 Relevance assessments

Relevance judgments are of critical importance to a test collection. For each topic it is necessary to compile a list of relevant documents — hopefully as comprehensive a list as possible. All TRECs have used the pooling method [22] to assemble the relevance assessments. In this method a pool of possible relevant documents is created by taking a sample of documents selected by the various participating systems.


```

<DOC>
<DOCNO>FT911-3</DOCNO>
<PROFILE>AN-BEOA7AAIFT</PROFILE>
<DATE>910514
</DATE>
<HEADLINE>
FT 14 MAY 91 / International Company News: Contigas plans DM900m east German
project
</HEADLINE>
<BYLINE>
By DAVID GOODHART
</BYLINE>
<DATELINE>
BONN
</DATELINE>
<TEXT>
CONTIGAS, the German gas group 81 per cent owned by the utility Bayernwerk, said
yesterday that it intends to invest DM900m (Dollars 522m) in the next four years
to build a new gas distribution system in the east German state of Thuringia. ...
</TEXT>
</DOC>

```

Figure 2: A document extract from the *Financial Times*.

This pool is then shown to the human assessors. The particular sampling method used in TREC is to take the top 100 documents retrieved in each submitted run for a given topic and merge them into the pool for assessment. This is a valid sampling technique since all the systems used ranked retrieval methods, with those documents most likely to be relevant returned first.

3.3.1 Overlap

The effect of pooling can be measured by examining the overlap of retrieved documents. Table 4 on page 9 summarizes the amount of overlap in the ad hoc and routing pools for each of the five TRECs. The first column in the table gives the maximum possible size of the pool. Since the top 100 documents from each run are judged, this number is 100 times the number of runs used to form the pool. The second column shows the number of documents that were actually in the pool (i.e., the number of unique documents retrieved in the top 100 across all runs) averaged over the number of topics. The percentage given in that column is the size of the actual pool relative to the possible pool size. The final column gives the average number of relevant documents in the pool and the percentage of the actual pool that was relevant.

Various tracks in TREC-4 and TREC-5 contributed documents to the ad hoc or routing pools. These are broken out in the appropriate rows within Table 4. The order of the tracks is significant in the table — a document retrieved in a track listed later is not counted for that track if the document was also retrieved by a track listed earlier.²

Since participants were allowed to submit two manual and two automatic ad hoc runs in TREC-5, many more ad hoc runs were judged than in previous years. The average actual pool size is also much larger than before. Much of the increase in the pool size can be attributed to the increase in the number of category B runs: while ad hoc runs in general increased from 40 to 77 between TREC-4 and TREC-5, category B runs increased from 6 to 16 runs. The comparatively large number of category B runs decreases overlap among runs in two ways. First, since category B runs retrieved documents from only the *Wall Street Journal* (WSJ) collection and the category A runs retrieved documents from seven different collections, the category B runs contribute many WSJ documents that

²The interactive track also contributed some documents to the ad hoc pool in TREC-5. However, the track used only a few topics, and many fewer than 100 documents were retrieved per topic. Table 4 does not include any interactive results for TREC-5.

<p><num> Number: 051</p> <p><dom> Domain: International Economics</p> <p><title> Topic: Airbus Subsidies</p> <p><desc> Description: Document will discuss government assistance to Airbus Industrie, or mention a trade dispute between Airbus and a U.S. aircraft producer over the issue of subsidies.</p> <p><narr> Narrative: A relevant document will cite or discuss assistance to Airbus Industrie by the French, German, British or Spanish government(s), or will discuss a trade dispute between Airbus or the European governments and a U.S. aircraft producer, most likely Boeing Co. or McDonnell Douglas Corp., or the U.S. government, over federal subsidies to Airbus.</p> <p><con> Concept(s):</p> <ol style="list-style-type: none"> 1. Airbus Industrie 2. European aircraft consortium, Messerschmitt-Boelkow-Blohm GmbH, British Aerospace PLC, Aerospatiale, Construcciones Aeronauticas S.A. 3. federal subsidies, government assistance, aid, loan, financing 4. trade dispute, trade controversy, trade tension 5. General Agreement on Tariffs and Trade (GATT) aircraft code 6. Trade Policy Review Group (TPRG) 7. complaint, objection 8. retaliation, anti-dumping duty petition, countervailing duty petition, sanctions
<p><num> Number: 168</p> <p><title> Topic: Financing AMTRAK</p> <p><desc> Description: A document will address the role of the Federal Government in financing the operation of the National Railroad Transportation Corporation (AMTRAK).</p> <p><narr> Narrative: A relevant document must provide information on the government's responsibility to make AMTRAK an economically viable entity. It could also discuss the privatization of AMTRAK as an alternative to continuing government subsidies. Documents comparing government subsidies given to air and bus transportation with those provided to AMTRAK would also be relevant.</p>
<p><num> Number: 207</p> <p><desc> What are the prospects of the Quebec separatists achieving independence from the rest of Canada?</p>

Figure 3: The evolution of TREC topic statements. Sample topic statement from TRECs 1 and 2 (top), TREC-3 (middle), and TREC-4 (bottom).


```

<num> Number: 251
<title> Exportation of Industry

<desc> Description:
Documents will report the exportation of some part of U.S. Industry to another
country.

<narr> Narrative:
Relevant documents will identify the type of industry being exported, the country
to which it is exported; and as well will reveal the number of jobs lost as a
result of that exportation.

```

Figure 4: A sample TREC-5 topic.

are not retrieved by category A runs. The second column of Table 5 shows the total number and the percentage of documents from each data source across all 50 ad hoc pools. Nearly half of the documents in the pools were from the *Wall Street Journal*. Second, the WSJ collection had a relatively small number of relevant documents, yet 100 documents per topic were added to the pools from each category B run. The third column of Table 5 shows the total number and the percentage of relevant documents from each data source across all 50 ad hoc pools. The WSJ collection contributed only 19% of the relevant documents. In general, pools for topics with fewer relevant documents exhibit less overlap, since systems that retrieve the same relevant documents often differ in the non-relevant documents they retrieve.

Table 4 also shows that the average number of relevant documents per topic has decreased over the years. As discussed below, NIST has deliberately chosen more tightly-focused topics to better guarantee the completeness of the relevance assessments. Larger pools coupled with fewer relevant documents means the percentage of the actual pool that is relevant has also been decreasing.

3.3.2 Quality of Relevance Assessments

Given the vital role relevance judgments play in a test collection, it is important to assess the quality of the judgments created using the pooling technique. In particular, both the *completeness* and the *consistency* of the relevance judgments are of interest. Completeness measures the degree to which all the relevant documents for a topic have been found; consistency measures the degree to which the assessor has marked all the “truly” relevant documents relevant and the “truly” irrelevant documents irrelevant.

The TREC-4 overview [9] reports on the results of an investigation of the completeness of the TREC-2 and TREC-3 relevance judgments. The relevance assessors judged the documents in new pools formed from the second 100 documents in the ranked results submitted by participants. On average, the assessors found approximately one new relevant document per run (i.e., one relevant document that was not in the pool created from the top 100 documents of each ranking). The distribution of the new relevant documents was roughly uniform across runs, but was skewed across topics — topics that had many relevant documents initially also had many more new relevant documents. This latter finding motivates the use of topics that have relatively few relevant documents in TREC, while the lack of bias against particular participants and the small number of new relevant documents found indicate the completeness is quite acceptable for TREC purposes.

A separate experiment to test the consistency of the relevance assessments was conducted after TREC-4. For each of the 49 TREC-4 ad hoc topics, a pool consisting of 200 randomly selected relevant documents (or all relevant documents if there were fewer than 200) and 200 randomly selected, judged nonrelevant documents was created. The pool was then given to two additional assessors who were each asked to judge the documents.

Of the 14,968 documents that were judged in this experiment, 71.7% received an unanimous judgment: 1992 (13.3%) unanimous relevant and 8742 (58.4%) unanimous nonrelevant. A three-way unanimous agreement is quite a stringent test, and these rates are somewhat higher than those found in other studies [10]. Nonetheless, there were areas of significant disagreement. On average, 30% of the documents that the primary assessor marked relevant were

Table 3: Topic length statistics by topic section. Lengths count number of tokens in topic statement including stop words.

	Min	Max	Mean
TREC-1 (51–100)	44	250	107.4
title	1	11	3.8
description	5	41	17.9
narrative	23	209	64.5
concepts	4	111	21.2
TREC-2 (101–150)	54	231	130.8
title	2	9	4.9
description	6	41	18.7
narrative	27	165	78.8
concepts	3	88	28.5
TREC-3 (151–200)	49	180	103.4
title	2	20	6.5
description	9	42	22.3
narrative	26	146	74.6
TREC-4 (201–250)	8	33	16.3
description	8	33	16.3
TREC-5 (251–300)	29	213	82.7
title	2	10	3.8
description	6	40	15.7
narrative	19	168	63.2

judged nonrelevant by both secondary assessors. In contrast, less than 3% of the documents judged nonrelevant by the primary assessor were considered relevant by both secondary assessors.

A primary goal of TREC is to construct test collections to facilitate IR research. In this context the important question regarding relevance assessments is not inter-assessor consistency per se but whether the assessments accurately reflect the relative merits of different retrieval techniques. Earlier studies have concluded that the ranking of retrieval techniques by effectiveness was stable across different sets of relevance assessments despite marked differences in the individual sets [14, 4]. While these conclusions are encouraging, the studies used small document collections and compared variants of the same retrieval systems, whereas TREC involves significantly larger collections and a wide variety of retrieval approaches. We therefore investigated how the ranking of systems by effectiveness varied with respect to different relevance assessment sets.

As a preliminary test of the stability of system rankings, we created five different “qrels” sets, where each qrels set consists of a particular set of assessments for each of the 49 topics. The original qrels set

Table 4: Overlap of submitted results

Ad Hoc			
	Possible	Actual	Relevant
TREC-1	3300	1279 (39%)	277 (22%)
TREC-2	4000	1106 (28%)	210 (19%)
TREC-3	2700	1005 (37%)	146 (15%)
TREC-4	7300	1711 (24%)	130 (08%)
ad hoc	4000	1345	115
confusion	900	205	0
dbmerge	800	77	2
interactive	1600	84	13
TREC-5	10,100	2671 (27%)	110 (04%)
ad hoc	7700	2310	104
dbmerge	600	72	2
NLP	1800	289	3

Routing			
	Possible	Actual	Relevant
TREC-1	2200	1067 (49%)	371 (35%)
TREC-2	4000	1466 (37%)	210 (14%)
TREC-3	2300	703 (31%)	146 (21%)
TREC-4	3800	957 (25%)	132 (14%)
routing	2600	930	131
filtering	1200	27	1
TREC-5	3100	955 (31%)	113 (12%)
routing	2200	854	94
filtering	900	100	19

Table 5: Number and percentage of documents in pool and relevant documents across all 50 ad hoc topics by document source

	Docs in Pool		Relevant Docs	
AP	16,460	(13%)	1644	(30%)
CR	10,467	(8%)	844	(15%)
FR88	5,341	(4%)	38	(1%)
FR94	8,347	(6%)	200	(4%)
FT	19,515	(15%)	1582	(29%)
WSJ	62,017	(47%)	1049	(19%)
ZIFF	9,601	(7%)	123	(2%)

Table 6: Mean average precision values for selected runs across five qrels sets

Run	Orig	A	B	\cup	\cap
uwgcl1	.2994	.2724	.2775	.3133	.2607
CLARTF	.2669	.2620	.2955	.3022	.2551
CLARTN	.2576	.2493	.2794	.2898	.2433
crnlAE	.2944	.2884	.2887	.3165	.2705
crnlAL	.2829	.2767	.2809	.3010	.2689
fsclt1	.1303	.1190	.1392	.1327	.1338
fsclt2	.1248	.1124	.1337	.1271	.1271

consists of the primary assessments for each topic — this is the qrels set released after TREC-4. The second and third qrels sets consist of a secondary relevance set for each topic. These qrels are equivalent to a qrels that might have been produced after a TREC conference if that set of assessors had been assigned those topics. Finally, we created a “union” qrels in which a document is considered to be relevant to a topic if any assessor judged it relevant to that topic, and an “intersection” qrels in which a document is considered relevant to a topic if all three assessors judged it relevant to that topic.

We evaluated each of the 33 TREC-4 category A ad hoc runs using each of the five qrels sets, and ranked the runs by decreasing mean average precision. Table 6 gives the mean average precision values for a subset of the 33 runs. Each pair of runs with similar names was submitted by a single participant. The final rankings produced by the different relevance assessments are very similar, though not identical. As was found in the earlier studies, in all cases where variants of a single system are compared, the ranking of the variants is the same across all qrels sets, even when the runs differ by a comparatively small margin. However, the ranking is somewhat less stable for different systems, even when the percentage difference in the original evaluation is considerably larger. For example, using the original qrels set, the uwgcl1 run is approximately 11% better than the CLARTF run, yet for the Set B qrels set, the CLARTF run is approximately 6% better than uwgcl1. Note that the neither the intersection nor the union qrels appear to be materially different from the original qrels. Indeed, the Set B qrels appears to differ the most from the original qrels set.

We intend to perform a more detailed analysis of how system rankings vary with the qrels set used. However, these preliminary results suggest that the TREC relevance assessments reliably measure re-

trieval effectiveness when comparing variants of the same system, and thus support the use of the TREC test collection as a research vehicle. Comparing effectiveness across systems is more difficult in that a single comparison is unlikely to be meaningful. As Tague-Sutcliffe and Blustein found in their analysis of the TREC-3 data [23], seemingly large differences in average effectiveness may not be statistically significantly different.

4 Evaluation

An important element of TREC is to provide a common evaluation forum. Standard recall/precision figures and some single evaluation measures have been calculated for each run and are shown in Appendix A. A detailed explanation of the measures is also included in the appendix.

Additional data about each system was collected that describes system features and system timing, and allows some primitive comparison of the amount of effort needed to produce the corresponding retrieval results. Due to the size of these system descriptions, they are not included in the printed version of these proceedings. The system descriptions are available on the TREC web site (currently <http://www-nlpir.nist.gov/trec>).

5 Retrieval Results

5.1 Introduction

One of the important goals of the TREC conferences is that the participating groups freely devise their own experiments within the TREC task. For some groups this means doing the routing and/or ad hoc task with the goal of achieving high retrieval effectiveness performance. For other groups, however, the goals are more diverse and may mean experiments in efficiency or unusual ways of using the data.

The overview of the results discusses the effectiveness of the systems and analyzes some of the similarities and differences in the approaches that were taken. In all cases, readers are referred to the system papers in this proceedings for more details.

5.2 TREC-5 ad hoc automatic results

The TREC-5 ad hoc evaluation used new topics (topics 251-300) against two disks of training documents (disks 2 and 4). A dominant feature of the ad hoc task was the desire of groups to continue work with the short topics like those used in TREC-4, but with

the luxury of being able to compare results with those from a longer or full topic, such as the topics in TREC-3. The systems doing automatic query building were required to submit at least one run using only the short version of the topic (the description field) to allow this comparison. Groups doing manual query building could use the full topic.

There were 77 sets of results for ad hoc evaluation in TREC-5, with 61 of them based on runs for the full (category A) data set. Of these, 32 used automatic construction of queries, and 29 used manual construction. Fourteen of the category B runs used automatically constructed queries, and two used manually constructed queries.

Figure 5 shows the recall/precision curves for the eight TREC-5 groups with the highest non-interpolated average precision using automatic construction of queries for the short version of the topics. The runs are ranked by the average precision and only one run is shown per group.

A brief summary of the techniques used in these runs shows the breadth of the approaches and the changes in approach from TREC-4. For more details on the various runs and procedures, please see the cited papers in this proceedings.

Cor5A2cr – Cornell (“Using Query Zoning and Correlation Within SMART: TREC 5” by Chris Buckley, Amit Singhal and Mandar Mitra) used the same term weighting scheme (Lnu.ltu) [19] developed for TREC-4, but worked on better query expansion techniques. This took two paths: a more careful selection of the pseudo-relevant documents for use in query expansion, and the use of both relevant and non-relevant documents to compute Rocchio weights. A smaller number of terms and phrases were added (25 terms, 5 phrases) than in TREC-4 (50 terms, 10 phrases). Breakdown of the results shows minimal improvement from the use of non-relevant documents in reweighting, but a 12% improvement from the use of a “query coverage” algorithm to more accurately pick the top 20 documents declared to be relevant. The run on the full topics using this same technique (*Cor5mle*) showed improved performance of 23% over using only the short version of the topics.

INQ301 – University of Massachusetts at Amherst (“INQUERY at TREC-5” by James Allan, Jamie Callan, Bruce Croft, Lisa Bellestros, John Broglio, Jinxi Xu and Hongmin Shu) took advantage of their inference net architecture

to combine the results of three different input queries for each topic. The first query was constructed in the same highly structured manner [2] as for the INQUERY TREC-4 queries. The second query contained only the most critical terms, and the third query used their local context analysis expansion method [24] to expand the initial query. Each of the three different types of queries performed better than using only the words in the short topic, with the largest improvement coming from the structuring of the query terms (24.5% improvement in average precision). Fusion of the three methods gave a nearly 40% improvement over using only the basic topic terms without structure and without expansion. The three methods perform differently at various recall levels, with the core query method performing the best at high recall (surprisingly) and the local context expansion performing the worst at the low recall (as would be expected). A similar run (revised) on the full topics (*INQ302c*) had 13.3% higher average precision than the run using only the short topics.

vtwnA1 – Apple Research Laboratories (“V-Twin: A Lightweight Engine for Interactive Use” by Daniel E. Rose and Curt Stevens) used an information access toolkit called V-Twin that was particularly built for use in interactive environments with very short queries. V-Twin is a vector-space model system using a variant of tf.idf weighting and length normalization. One aspect of this engine is its minimal memory requirement and very low indexing overhead: the index for TREC-5 was only 22.5% of the text size. A second aspect is the use of a new weighting function, called SQR, that is especially designed for interactive use with short queries. This function showed useful results during training from the TREC-4 data, but was inadvertently not used in the official TREC-5 results. The TREC-5 results did use automatic relevance feedback similar to other systems. Later unofficial runs showed that the SQR weighting function did not actually help performance, but more importantly, retained its more user-intuitive weighting without hurting performance.

pircsAAS – Queens College, CUNY (“TREC-5 English and Chinese Retrieval Experiments using PIRCS” by K.L. Kwok and L. Grunfeld) used a special term weighting scheme developed for short queries after TREC-4 [12] that is a way of

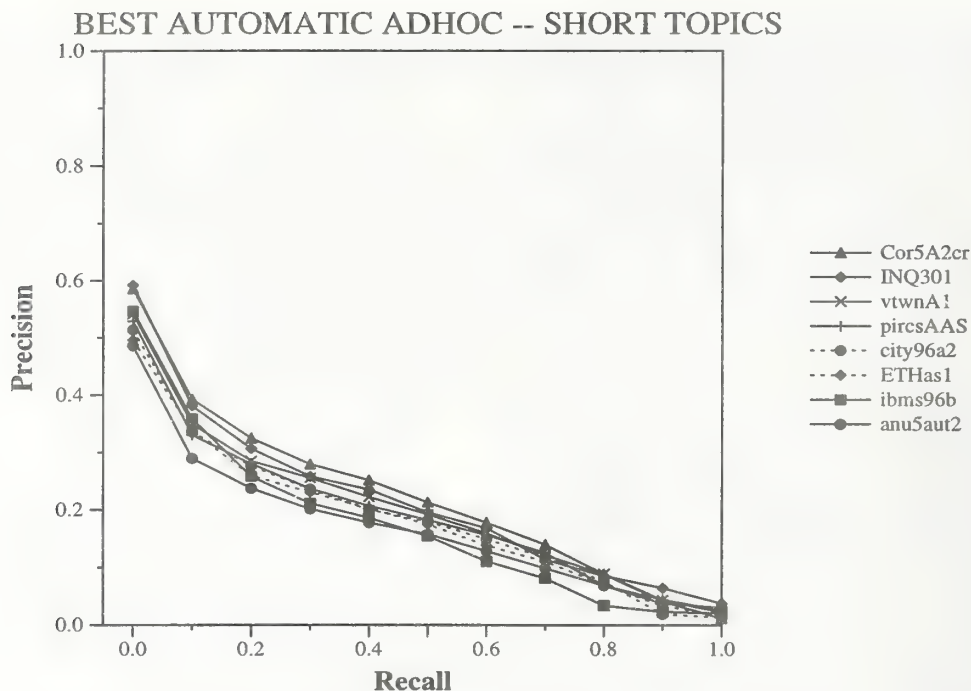


Figure 5: Recall/Precision graph for the top eight ad hoc, automatic, short runs.

simulating the manual reweighting of terms done in TREC-4. Also new for TREC-5 was an experiment in building 2-word phrases that are concatenations of high frequency and low frequency query terms to help match statistically-produced phrases in the documents. As in TREC-4, 50 expansion terms are picked from the top-ranked 40 subdocuments. The new term weighting scheme gave an almost 15% improvement for the short topics, with expansion adding an additional 13%. The experiment in phrases did not work and hurt performance by 6%. When these techniques were used on the full topics (*pircsAAL*), there was a 36% improvement in average precision over the run for short topics.

city96a2 – City University, London (“Okapi at TREC-5” by M.M. Beaulieu, M. Gatford, Xiangji Huang, S.E. Robertson, S. Walker, and P. Williams) used essentially the same OKAPI weighting and expansion schemes [17, 18] used in TREC-4. An emphasis was placed on efficiency this year, resulting in runs at 4 times the TREC-4 speeds. Additionally many experiments were tried (but failed), including variations of the term expansion algorithms and several experiments with adjacent term pairs. The *city96a2* run used the top 15 documents to get additional terms, for a total of 30 terms per query. Unlike

the groups mentioned earlier, City experimented with the full topic, and then ran the short topic using a variation of the best techniques. The full topic run used the top 30 documents to get additional terms, and had a maximum of 55 terms per query. The full topic results showed a 30% improvement over the short version of the topic.

ETHas1 – Swiss Federal Institute of Technology (ETH) (“SPIDER Retrieval System at TREC-5” by Jean-Paul Ballerini, Marco Büchel, Ruxandra Domenig, Daniel Knaus, Bojidar Mateev, Elke Mittendorf, Peter Schäuble, Páraic Sheridan, and Martin Wechsler) used the basic Cornell TREC-4 term weighting schemes (*Lnu.ltn*), but a somewhat different feature selection method for picking the top 50 terms and 20 phrases. The full topic results showed a 40% improvement over the short version of the topic.

ibms96b – IBM T.J. Watson Research Center (“TREC-5 Ad Hoc Retrieval using K Nearest-Neighbors Re-Scoring” by Ernest P. Chan, Santiago Garcia and Salim Roukos) built a two-pass retrieval system using the OKAPI weighting formula for a first scoring/ranking of the documents, and then a second pass algorithm that rescored the top 1000 documents using a combination of the first score and a score based on us-

ing the top 10 documents as queries. They used single terms plus statistical two-word phrases, and a new sigmoidal suppression factor for length normalization.

anu5aut2 – Australian National University (“ANU/ACSys TREC-5 Experiments” by David Hawking, Paul Thistlewaite and Peter Bailey) used a parallel architecture with an emphasis on efficiency. The automatic queries were generated by adding multi-word terms to the original words, using statistical co-occurrence to locate these multi-word terms. Term frequency weights were used in the ranking algorithm, replacing the semantics-based techniques used in TREC-4.

Shortly before this overview was finalized, NIST was notified that the Lexis-Nexis run that had previously been categorized as automatic actually had some (minimal) human intervention. This run has been removed from Figure 5, but will be included in this section for discussion to ease the confusion and because the techniques are more related to the automatic techniques than to the other manual runs.

LNaDesc2 – Lexis-Nexis (“Ad Hoc Experiments Using EUREKA” by Allan Lu, Maen Ayoub and Jianhua Dong) used their experimental toolbox, the EUREKA system, for experiments to improve the ranking of their top set of retrieved documents. This was done to improve performance at the high precision end of the performance curve (of importance for interactive systems), and also to improve the pseudo-relevance feedback necessary for query expansion. They experimented with three complementary techniques: 1) use of inter-term distance between nouns in the short version of the topic, 2) fusion (using logistic regression) of the results from three different ranking measures, and 3) clustering of documents in the top 20 documents initially retrieved in order to improve accuracy.

Three experimental themes dominate these top groups. The first is the continued investigation into query expansion. All groups except ANU used the top n documents/subdocuments/passages to pick m terms, where n ranged from 15 to 40, and m ranged between 25 and 70 (not all groups identified these numbers). There were interesting variations in this theme. For example, Lexis-Nexis clustered the top documents in hopes of finding better features, INQUERY used noun groups based on their occurrence in the top ranked passages, and IBM used the top 10 documents as individual queries. Cornell tried some

experiments using negative weights, but these were not successful. ANU expanded the terms in the short topics by using statistically co-occurring terms across the corpus, rather than using the top n documents as a source of expansion terms. As can be seen, differing amounts of improvement were found for these techniques, but the wide range of performance improvements is likely to come from interaction with the other features in the underlying systems rather than the particular expansion technique used.

Most (but not all) of the TREC-5 ad hoc experiments used some type of query expansion. Of particular note are two groups that tried unique methods. The first group, Open Text Corporation (“Experiments with TREC using the Open Text Livelink Engine” by Larry Fitzpatrick, Mei Dent, and Gary Promhouse) gathered terms for expansion by looking at relevant documents from past topics that were loosely similar to the TREC-5 topics. This worked very well, although their results were lowered by further shortening the topics to “imitate” real user requests. (For further work by Open Text on this method see [7].) The second group, the University of Kansas (“Corpus Analysis for TREC 5 Query Expansion” by Susan Gauch and Jianying Wang), used a complicated corpus linguistics approach involving context vectors and mutual information values. This approach was not as successful, likely because of the number of parameters that had to be discovered and tuned.

The second experimental theme in the TREC-5 ad hoc runs is the growing interest in getting more information from the initial topic, even the short version of the topic. Most of the top groups tried various schemes to improve on the “bag of words” approach to basic topic processing. INQUERY used their elaborate automatic query structuring method that has been enhanced over the years, with an improvement over the baseline query of 24.5%. Lexis-Nexis used a distance relationship between nouns in a topic to improve term weighting, and PIRCS used an automatic reweighting scheme on key concepts in the topic to gain 15% performance. Cornell used an analysis of the match between key concepts in the topic and those in small windows of the top 50 documents to rerank these initial documents (12% improvement). Note that these schemes not only improve performance in the initial ranked list (critical to interactive system performance), but also improve the set of documents used for query expansion (and therefore the high recall performance). More intense work with the TREC topics is a theme that is likely to expand in TREC-6.

The third experimental theme is the continued interest in data fusion. Three category A groups tried major data fusion experiments. The Lexis-Nexis group tried fusing results from three different ranking algorithms (variations on the OKAPI algorithm and the Cornell algorithms). The INQUERY system used fusion results from three different queries: a basic structured query, a "key concepts" query, and an expanded query. Experiments were also done by RMIT ("The MDS Experiments for TREC5" by Marcin Kaszkiel, Phil Vines, Ross Wilkinson and Justin Zobel) combining results from various parts of the original topic (and its expansion) when used as input to various cosine and OKAPI measures.

Two category B groups also experimented with data fusion. The Universite de Neuchatel ("Report on the TREC-5 Experiment: Data Fusion and Collection Fusion" by Jacques Savoy, Anne Le Calvé, and Dana Vrajitoru) tried fusion of results from multiple weighting algorithms (OKAPI and several different SMART weighting schemes), using logistic regression to create the final ranks. The University of California, San Diego ("Using Relevance to Train a Linear Mixture of Experts" by Christopher Vogt, Garrison Cottrell, Richard Belew and Brian Bartell) investigated the mixing of results from three "experts." These experts consisted of two weightings of SMART (binary and tf.idf), and the LSI techniques, and their experiments concentrated on examining the effects of various training techniques.

The experimental work in TREC-5 using the short version of the topic for automatic query construction shows considerable progress over that done in TREC-4. Some of this growth is due to more experience with (and acceptance of) the shorter topics, but much of it is due to accumulated knowledge in areas like query expansion and data fusion. Query expansion using the top-retrieved documents was started in TREC-3 by several groups, and by TREC-5 most groups have devised variations suitable for their particular systems. Data fusion experiments became more common, both in the ad hoc task and in the routing task. New for TREC-5 was more work with the initial topic.

Figure 6 shows the comparison of results between using only the short version of the topic (the description) and using the full topic, where both sets of results are shown for four of the systems previously described. Note that for all four of the systems there is a sharp rise in performance using the full topic as opposed to using the short topic only – PIRCS goes up by 36%, ETH by 40%, City by 30%, and the V-Twin Apple run by 9%. With the exception of the

Apple runs, this performance difference appears at all levels of recall, i.e., there is a 25 to 30% difference even looking only at the top 10 documents retrieved.

Several reasons have been given for the generally large improvement in performance using the full topics. These center around two main causes: improved statistical power from the additional words, and more term variations or expansions in the longer topics. The increased statistical power of the full topic has many subtle effects, including improved term frequency information to give more weight to some terms, and more syntactic information to produce better phrases. The addition of term variations and expansions provides not only useful synonyms, but also "corrections" to stemmers and stopword lists via redundant words.

It is interesting to note that the Apple runs show the least difference, and the Lexis-Nexis run on the short form of the topics demonstrates potential ability to perform well automatically using minimal input. These commercial groups have clearly adapted to short user topics, and this adaptation is an important research area. Not only will the TREC-6 topics contain both a short and full version, but an even shorter title version (on the order of three words) will also be added.

TREC has not particularly emphasized efficiency, although some minimal efficiency is needed just to search the large text collections. Several groups, including RMIT [16], ANU [11], and George Mason University, have examined efficiency issues in past TRECs. The group from George Mason University ("Using Relevance Feedback within the Relational Model for TREC-5" by David Grossman, Carol Lundquist, John Reichart, David Holmes, Abdur Chowdhury and Ophir Frieder) has based their work on using efficient parallel database systems, and for TREC-5 investigated methods of incorporating relevance feedback into SQL. Two groups specifically investigated efficiency algorithms for TREC-5. The Computer Technology Institute ("Parallel Techniques for Efficient Searching over Very Large Text Collections" by B. Mamalis, P. Spirakis, and B. Tampakas) reported on various experiments using a new parallel version of their VSM-based traditional system. Dublin City University ("TREC-5 Experiments at Dublin City University: Query Space Reduction, Spanish and Character Shape Encoding" by Fergus Kelledy and Alan F. Smeaton) experimented with reducing the number of terms to be processed for each query by using several thresholding techniques. Both these groups were able to enhance efficiency without sacrificing effectiveness.

BEST AUTOMATIC ADHOC -- SHORT VS LONG TOPICS

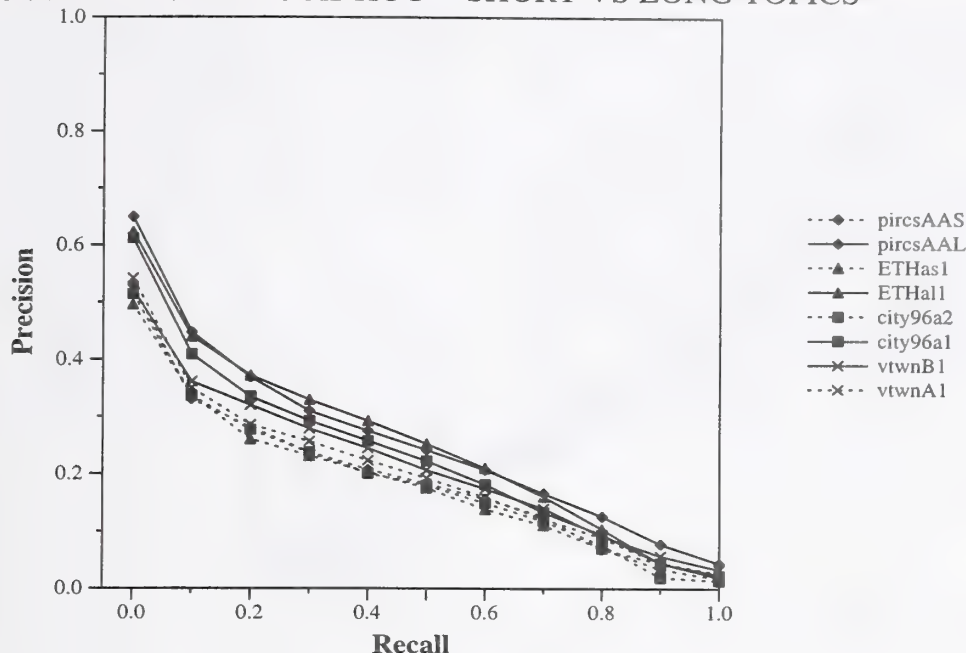


Figure 6: Comparison of short vs. long topics for selected systems.

5.3 TREC-5 ad hoc manual results

Figure 7 shows the recall/precision curves for the eight TREC-5 groups with the highest non-interpolated average precision using manual construction of queries. Note that manual query construction included user interaction in TREC-5, i.e., the rules were modified so that initial results could be viewed and the queries changed, with no restrictions on how much time could be spent. Therefore the amount of human effort required for these various techniques should be considered when comparing the retrieval results. A short summary of the techniques used in these runs follows; for more details on the various runs and procedures, see the cited papers in this proceedings.

ETHme – Swiss Federal Institute of Technology (ETH) (“SPIDER Retrieval System at TREC-5” by Jean-Paul Ballerini, Marco Büchel, Ruxandra Domenig, Daniel Knaus, Bojidar Mateev, Elke Mittendorf, Peter Schäuble, Páraic Sheridan, and Martin Wechsler) did a completely manual search operation, including manual construction of the queries and query expansion with all found “relevant” documents. The users (who were information science students) had no time constraints, and spent about 30 to 40 minutes per topic. The basic retrieval system used

was the same as for the automatic ad hoc runs, with an improvement in performance of 31% over the automatic run using the full topics.

uwgex1 – University of Waterloo (“Interactive Substring Retrieval (MultiText Experiments for TREC-5)” by Charles L.A. Clarke and Gordon V. Cormack) used queries that were manually built in a special query language called GCL. This query language uses Boolean operators and proximity constraints to create intervals of text that satisfy specific conditions, and the concentration of these intervals of text is used to produce the ranking. The TREC-5 experiments involved several ranking method trials, including one with length normalization. The main experiment, however, tested the interactive use of the system to find suitable expansion terms.

LNmFull2 – Lexis-Nexis (“Ad Hoc Experiments Using EUREKA” by Allan Lu, Maen Ayoub and Jianhua Dong) repeated their *LNDesc2* run using manually-edited versions of the full topic. The inter-term distance between nouns was not used because it was specifically built for the short version of the topics. No interaction was involved; results from the manually-edited queries were submitted without query modification. The manually-edited queries showed a 9% improve-

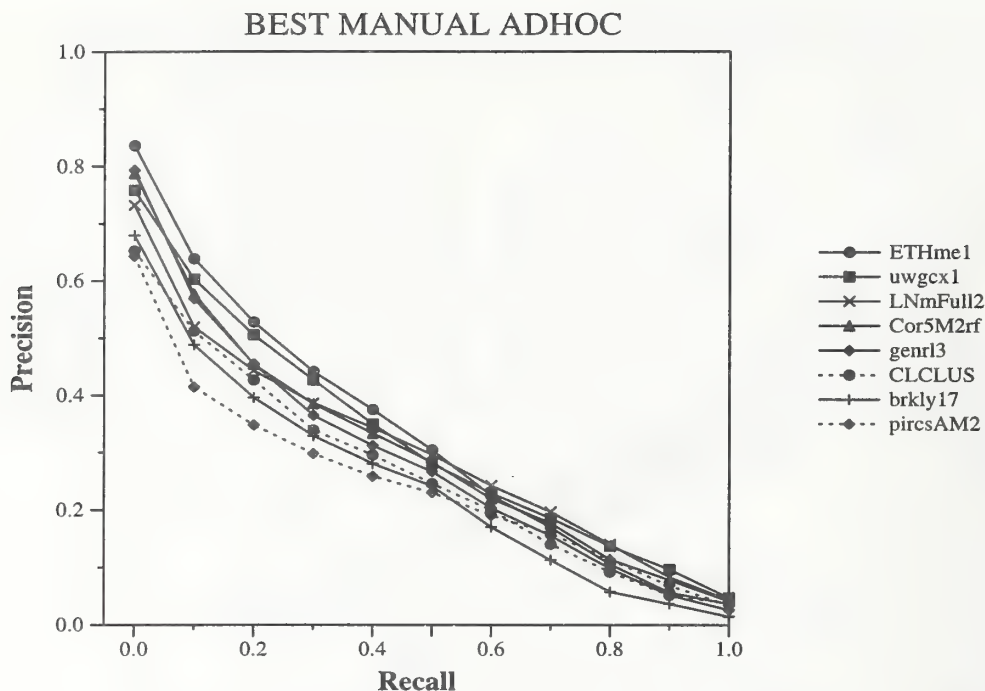


Figure 7: Recall/Precision graph for the top 8 ad hoc, manual runs.

ment over using only the short version of the topics.

Cor5M2rf – Cornell (“Using Query Zoning and Correlation Within SMART: TREC 5” by Chris Buckley, Amit Singhal and Mandar Mitra) used the same methods as the automatic runs with the full topics, but made manual relevance judgments on the top documents before using these documents for relevance feedback. On average only about 5 minutes was used to judge 25 documents per topic, and the improvement in performance was 15% over using the full set of top documents.

genrl3 – GE / Lockheed Martin / NYU / Rutgers (“Natural Language Information Retrieval: TREC-5 Report” by Tomek Strzalkowski, Louise Guthrie, Jussi Karlgren, Jim Leistensnider, Fang Lin, Jose Perez-Carballo, Troy Straszheim, Jin Wang and Jon Wilding) continued their investigations into contributions of natural language processing. This particular run represents experiments with users finding phrases and sentences to add to the initial query based on the top 10 documents retrieved from the initial queries (5 to 10 queries used per topic). These manually-expanded queries were run through the natural language processing modules to generate the fi-

nal results.

CLCLUS – CLARITECH Corporation (“CLARIT Compound Queries and Constraint-Controlled Feedback in TREC-5 Ad-Hoc Experiments” by Nataša Milić-Frayling, Xiang Tong, Chengxiang Zhai and David A. Evans) used the commercial version of the CLARIT system in a series of experiments comparing the use of different sources for manual query expansion and the use of Boolean constraints to improve input to that expansion. This particular run used manual editing of an automatically-generated initial query, manual additions of Boolean constraints to that query, and finally manual expansion of the constrained query using terms selected from terminology clusters built from the full corpus. The results were not very different from the second manual CLARIT run, which used manually selected terms from the retrieved top documents.

Brkly17 – University of California, Berkeley (“Term importance, Boolean conjunct training, negative terms, and foreign language retrieval: probabilistic algorithms for TREC-5” by Fredric C. Gey, Aitao Chen, Jianzhang He, Liangjie Xu and Jason Meggs) used manually-reformulated queries based on both examination of the top retrieved documents and expansion using the

News database of the MELVYL electronic catalog (similar to that done in TREC-3 [6]. Experiments using negatively weighted terms showed a 15% decrease in performance. The basic retrieval system is a logistic regression model [5] that combines information from six measures of document relevancy based on term matches and term distribution. The coefficients were learned from the training data.

pircsAM2 – Queens College, CUNY (“TREC-5 English and Chinese Retrieval Experiments using PIRCS” by K.L. Kwok and L. Grunfeld) repeated their manual TREC-4 experiment to contrast with the their automatic simulation of the manual work. This run is the result of both a manual term reweighting, and a manual expansion of up to three terms per topic. There was no modification of the query based on examining the results. The reweighting part contributed about a 15% improvement over no reweighting, but less than a 1% improvement over the automatic version of this. The manual expansion did not show improvements over the automatic expansion.

There has been an interesting evolution in the methods used for manual query construction over the various TRECs. The rich topics for TRECs 1 and 2 showed little difference in performance for manually-produced queries over the automatic runs. When the concept section was removed from topics starting in TREC-3, most groups tried manual runs as a way of improving their scores. Examples of this would be systems such as INQUERY and PIRCS, where manual editing, manual reweighting and minimal manual query expansion were done to produce manual versions of their automatic runs. This was even more pronounced in TREC-4, where groups generally feared poor results from the short topics.

In general these low-effort manual runs were not done in TREC-5. The *pircsAM2* run was a repeat of their TREC-4 manual reweighting experiment, but was done in TREC-5 to verify that the automatic version of this had successfully replaced the manual version (which it had). Groups seemed more comfortable with the automatic version of their initial queries, and with automatic expansion, and felt no need for manual edits.

This trend was reinforced by the change of rules allowing unrestricted interaction with the systems in TREC-5. Except for the *pircsAM2* and *LNmFull2* runs, all runs performed some type of interaction. The simplest was the *Cor5M2rf* run, where minimal manual effort was spent to determine the “rel-

evant” documents for use in the relevance feedback rather than automatically taking the top 20 documents. This can be contrasted with the *ETHme1* run, where students spent between 30 and 40 minutes per topic producing the “perfect” query.

The manual runs for TREC-5 can be loosely classified into three categories. The first category, exemplified by *uwgcz1* and *Brkly17*, used queries completely manually generated using some type of auxiliary information resource such as online dictionaries (*uwgcz1*) or news databases (*Brkly17*). The query generated for *uwgcz1* uses Boolean-type restrictors, whereas the query generated for *Brkly17* uses natural language. In both cases, the results from initial retrievals were then further expanded by looking at the retrieved documents, similar to the ways in which users of these systems might modify queries to get more relevant documents.

The second category of manual query construction runs involves a more complex type of human-machine interaction. The *CLCLUS* run is a result of experiments examining a multi-stage process of query construction, where the goal is to investigate better sets of tools that allow users to improve their queries (see [15] for similar experiments in TREC-4). The CLARITECH group tried using different sources for suggestions of expansion terms and also various levels of user-added constraints to the expansion process.

Two other groups also investigated human-machine interaction, with an emphasis on relevance feedback. FS Consulting (“Document Retrieval Using the MPS Information Server (A Report on the TREC-5 Experiment)” by Francois Schiettecatte) started with a baseline of manual queries, and then added terms based either on the user’s selection of two relevant documents, or the top two system-selected documents. The University of North Carolina (“An Investigation of Relevance Feedback using Adaptive Linear and Probabilistic Models” by Robert Sumner and W.M. Shaw) tested two different models of relevance feedback, as applied by two different users.

By far the largest category of runs, however, could be labelled as “manual exploration” runs. This was specifically stated by GE, where the goal was to ask users to pick out phrases and sentences from the retrieved documents to add to the query, in hopes that this process can be imitated by automatic methods. Similar reasons are likely to apply to the ETH run and the Lexis-Nexis run, with the Cornell run and PIRCS run being more specific versions of exploration (performance differences using “relevant” documents vs the top 20 and performance differences using manual as opposed to automatic reweighting of terms).

Table 7: Variations of the hardness measure for different TRECs.

	TREC-3	TREC-4	TREC-5
best	0.6285	0.5657	0.5199
average	0.3767	0.2814	0.2460
median	0.3692	0.2929	0.2049

This use of the manual query construction category to identify new automatic methods is particularly promising.

5.4 Comments on the TREC-5 ad hoc topics

In general the TREC-5 ad hoc topics were thought to be more difficult than the TREC-4 ad hoc topics. The Cornell paper (“Using Query Zoning and Correlation Within SMART: TREC 5” by Chris Buckley, Amit Singhal and Mandar Mitra) contains a table (Table 16) showing comparison of the average precision of the Cornell runs over the five TRECs. Of particular interest here is the fact that the TREC-5 Cornell system performed about 34% worse on the TREC-5 topics than on the TREC-4 topics. Whereas some of this difference has to do with training, most of the difference is due to “harder” topics.

To further examine this issue, a measure of “hardness” was revived from earlier experimental use in TREC-2. The hardness measure is defined as an average over a given set of runs of the precision for each topic after all relevant documents have been retrieved *OR* after 100 documents have been retrieved, if more than 100 documents are relevant. This measure is therefore oriented towards high recall performance and how well systems do at finding all the relevant documents.

The hardness measure can be calculated over different sets of runs, using different types of averaging. Table 7 shows three types of averages for three TRECs. All these averages are for the category A ad hoc runs, both automatic and manual, and both full and short versions of the topic. The row labelled “best” shows the averages across all 50 topics of the best results for each topic by any run. This is therefore the highest possible performance. The row labelled “average” is the mean across all runs for all 50 topics, and the “median” row is the corresponding median performance.

As can be seen, the topics have grown progressively harder (the lower the hardness number, the harder the task). The drop between TREC-3 and TREC-4

was expected since the TREC-3 results are all based on the full topic and the TREC-4 results are based only on the short version of the topic. The TREC-5 results include both full and short versions of the topic, but compared to TREC-3, show a 35% drop in performance (using the average measure). This was unexpected, both by NIST and by the participants.

The drop in performance on the TREC-5 topics occurred not only at the high recall end of performance (as measured by the hardness measure) but also at the high precision end (at 30 documents retrieved). Appendix B, “Summary Performance Comparisons TREC-2, TREC-3, TREC-4, TREC-5” by Karen Sparck Jones, illustrates this, and also discusses some of the issues involved in these comparisons.

Investigation has been started at NIST into why the TREC-5 topics are more difficult. Table 8 shows a first attempt to isolate factors associated with the hardness measure. The topics are sorted by hardness, using the average hardness shown in Table 7. The second column contains the number of relevant documents for each topic, and the third column the length of the topic (unstemmed, without stopword removal). There appears to be little correlation between the hardness and the number of relevant documents or the topic length. A correlation coefficient using the Pearson product moment gives a correlation of 0.19 between the number of relevant documents and the hardness, and a correlation of 0.14 between the topic length and the hardness. This can be compared with a correlation of 0.20 between the topic number and the hardness, which is clearly a random correlation.

The fourth column of Table 8 shows the hardness of the topic as computed using all runs, while the fifth and sixth columns show the hardness as calculated separately for the automatic systems and the manual systems. Some of the topics show large differences for these runs, such as topic 293. One reason that this might happen is that automatic systems could not construct as accurate a query as the manual systems, as looks to be the case in topic 263. But another hypothesis is that the manual systems are able to find (and rank in the top 100) documents that the automatic systems could not. This led to an investigation of which systems found which relevant documents, and of particular note is the fact that large numbers of unique documents (those only found by one group) occur in TREC-5. The final columns of Table 8 are the percentage of the relevant documents that were unique, both for all runs, and also looking at the unique relevant documents found only by the manual runs. There is a correlation coefficient of 0.33

Table 8: Correlation between hardness and topic characteristics

Topic	# Rel	Topic Length	Hardness All	Hardness Auto	Hardness Manual	% Unique	% Manual Only
281	1	115	0.0000	0.0000	0.0000	0.0	0.0
296	1	53	0.0000	0.0000	0.0000	0.0	0.0
267	4	97	0.0000	0.0000	0.0000	50.0	0.0
293	41	78	0.0484	0.0252	0.0740	29.3	34.1
292	59	92	0.0514	0.0191	0.0871	35.6	37.9
268	45	117	0.0576	0.0424	0.0743	22.2	11.1
278	7	44	0.0609	0.0223	0.1035	14.3	28.6
275	19	100	0.0638	0.0592	0.0690	21.1	5.3
255	109	81	0.0782	0.0597	0.0986	41.3	33.0
252	37	53	0.1024	0.0946	0.1109	21.6	13.5
300	44	84	0.1032	0.0675	0.1426	20.5	25.0
290	119	58	0.1115	0.0656	0.1621	26.1	32.8
256	22	67	0.1289	0.1577	0.0972	22.7	13.6
291	407	100	0.1480	0.0969	0.2045	57.2	41.8
279	2	88	0.1557	0.1094	0.2069	0.0	0.0
299	62	66	0.1579	0.1472	0.1696	12.9	11.3
264	281	76	0.1608	0.1341	0.1903	48.0	37.1
284	70	78	0.1761	0.1603	0.1936	25.7	21.4
295	15	73	0.1803	0.1583	0.2046	20.0	13.3
287	40	134	0.1844	0.1727	0.1974	20.0	20.0
263	15	123	0.1847	0.1250	0.2506	0.0	0.0
282	131	30	0.1882	0.1709	0.2072	48.1	47.3
260	22	94	0.1923	0.1392	0.2508	4.5	13.6
294	160	77	0.1969	0.0912	0.3134	24.4	43.1
283	84	81	0.2004	0.2184	0.1806	13.1	10.7
289	141	163	0.2049	0.1647	0.2493	17.7	12.8
266	139	36	0.2125	0.1763	0.2524	27.3	52.5
254	85	83	0.2170	0.1761	0.2621	24.7	24.7
258	115	96	0.2184	0.1441	0.3003	18.3	19.1
251	579	50	0.2195	0.1000	0.3514	56.6	46.5
271	86	49	0.2326	0.1697	0.3019	9.3	9.3
298	91	47	0.2365	0.2236	0.2508	19.8	9.9
269	594	70	0.2459	0.1397	0.3631	63.1	58.9
272	36	65	0.2495	0.2309	0.2701	2.8	2.8
277	74	48	0.2497	0.2166	0.2861	21.6	12.2
297	86	142	0.2735	0.1457	0.4146	4.7	5.8
257	135	48	0.2826	0.2869	0.2779	14.8	11.9
261	87	214	0.3122	0.3057	0.3195	11.5	10.3
286	142	57	0.3462	0.3622	0.3286	16.9	13.4
270	116	156	0.3543	0.2703	0.4469	17.2	14.7
288	92	109	0.3770	0.3974	0.3546	3.3	4.3
274	119	38	0.3815	0.2772	0.4966	12.6	13.4
262	4	99	0.4508	0.3281	0.5862	0.0	0.0
259	36	63	0.4982	0.4991	0.4971	2.8	0.0
280	32	47	0.5435	0.5293	0.5593	0.0	0.0
285	261	140	0.5684	0.5566	0.5814	11.5	9.6
253	10	92	0.5869	0.5687	0.6069	0.0	0.0
273	513	69	0.6349	0.6247	0.6462	33.7	23.6
276	7	89	0.7143	0.7321	0.6946	0.0	0.0
265	147	56	0.7572	0.8047	0.7048	12.9	12.2

between the percentage of unique relevant documents for a topic and its hardness measure.

Having large numbers of unique documents for a given topic means that it is harder for the systems to retrieve all the relevant documents, and since this is what the hardness measure is based on, it is not surprising that there is a correlation between the two measures. However it is not obvious whether the large number of unique relevant documents is causing the scores to be lower, or whether the topics are so difficult that there are many relevant documents that can be found only by one system.

Figures 8 and 9 show two additional breakdowns of the sources of the relevant documents. Figure 8 shows the percentages contributed by each type of ad hoc run (and some tracks) to the judgment pool, and to the relevant documents. For example, whereas 35% of the pool for relevance judgments came from the automatic systems, only 6% of the unique relevant documents were found by these systems. The manual systems contributed 23% of the pool, and 29% of the unique relevant documents. This could mean either that these documents were simply not retrieved by the automatic systems in general, or that the automatic systems ranked them lower than nonrelevant documents.

Figure 9 gives a different view of the same issue by looking at the systems that retrieved the most unique relevant documents. It is noteworthy that most of this contribution involves manually produced results, although systems using "unusual" methods, such as the proximity ranking done by ANU and Waterloo, are finding somewhat more unique relevant documents than manual versions of more traditional systems such as ETH and Berkeley. Further investigation into the patterns of unique relevant documents in past TRECs is needed to help resolve these questions.

Obviously there are other factors as to why the TREC-5 topics seem to be harder. One such factor lies in the construction of the topics themselves, but this is difficult to measure objectively. A brief examination of the topics for TRECs 3, 4 and 5 suggests that the TREC-3 topics were more simplistic. People familiar with the workings of search engines are likely to be able predict which of these topics will be "easy", and which will not. The topics in TREC-4 and 5 appear to be more complex, i.e., they are asking for very specific information about multiple concepts rather than general information about a single area of interest.

These observations are consistent with the evolution of topic construction at NIST. The TREC-3

ad hoc topics (151–200) were the first topics "built" by the relevance assessors; earlier topics were constructed outside of NIST. The instructions for TREC-3 asked the relevance assessors to bring in some initial areas of interest (called "seeds"), and then they searched the document collections in order to create the topics. Because this was a somewhat artificial way of building topics, the TREC-4 ad hoc topics (201–250) were completely written before any interaction with the documents, and topics were selected based on the likely number of relevant documents that would be found by the systems (to eliminate very broad or very narrow topics). This same procedure was used in TREC-5, but by then the assessors were more experienced at building topics and may have built more difficult ones.

As a final comment, it is likely that the observed difficulty of the TREC-5 topics is due to a combination of factors. Even though the topics themselves are more complex, it is not easy to predict which topics will be difficult. There are interactions between the subject domains of the topics and the documents being searched, between the topics and the relevance assessments, and finally between the topics and the methodologies of query expansion and ranking of documents being used by the systems. More knowledge is needed about what makes some topics more difficult than others, especially if it can lead to what types of tools or systems are more appropriate for certain types of topics.

5.5 TREC-5 routing results

The routing evaluation used a specifically selected subset of the training topics against a new set of test documents. In TREC-4 there was difficulty obtaining new data for testing; the outcome was performing routing tests using the *Federal Register* (with new data) for 25 of the topics, and using training data and "net trash" for testing the other 25 topics. This situation was clearly not ideal and for TREC-5 NIST held back decisions on the routing topics until a new data source could be found.

When the FBIS data described earlier became available, it was decided to pick topics that had many relevant documents in the *Associated Press* data, on the assumption that the FBIS data would be similar to AP. Because of delays in getting and processing the data, this assumption could not be checked out, and problems arose that will be discussed later.

There were a total of 26 sets of results for routing evaluation, with 23 of them based on runs for the full data set. Of the 23 systems using the full data set,

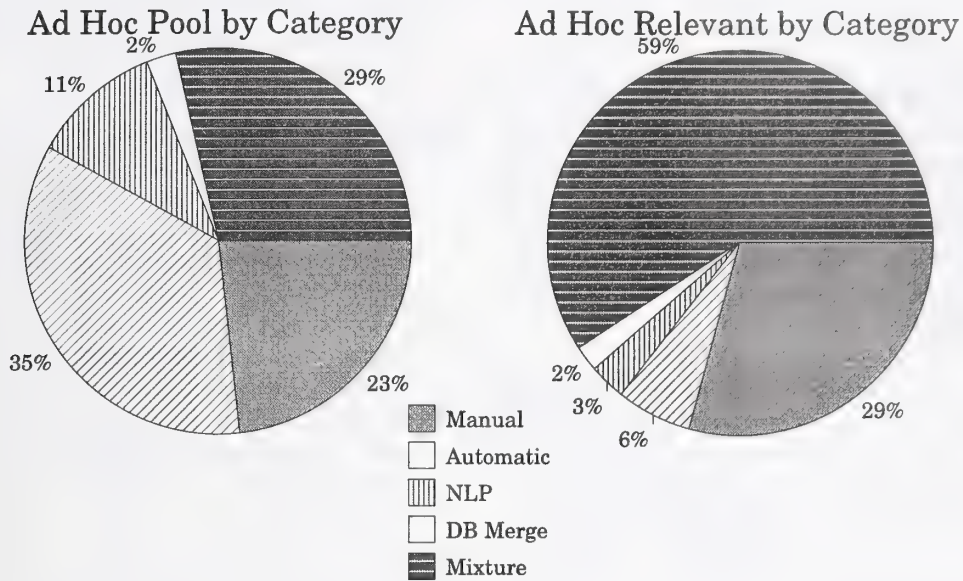


Figure 8: Distribution of categories in judged and relevant document pools.

Unique Contribution to Ad Hoc Relevants

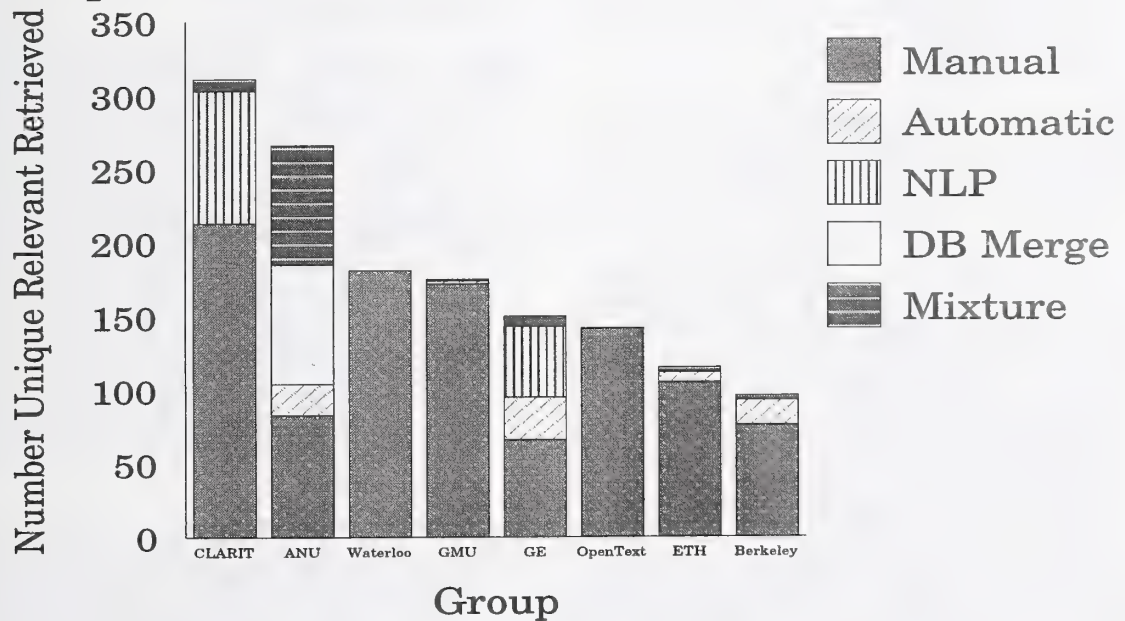


Figure 9: Percentage of unique relevant documents by category for groups retrieving many unique relevant documents.

21 used automatic construction of queries, and 2 used manual construction. There were 3 sets of category B routing results, all using automatic construction of queries.

Figure 10 shows the recall/precision curves for the eight TREC-5 groups with the highest non-interpolated average precision for the routing queries. The runs are ranked by the average precision based on the 39 topics that had more than five relevant documents.³ A short summary of the techniques used in these runs follows. For more details on the various runs and procedures, please see the cited papers in this proceedings.

city96r2 – City University, London (“Okapi at TREC-5” by M.M. Beaulieu, M. Gatford, Xiangji Huang, S.E. Robertson, S. Walker, and P. Williams) expanded on their query term selection method used in TREC-3 [18] (the predecessor of the Dynamic Feedback Optimization (DFO) algorithm developed by Cornell [3]). To combat overfitting of the training data, they divided the data into 3 partitions, using one to select the initial pool of terms, a second partition to do a final selection of the terms, and the third as an evaluation test set. After their “best” method was selected, the final queries were built using half the training data to select the initial pool, and the second half to do the final selection. Additionally they experimented with merging results from queries that used different term selection methods.

Cor5R1cc – Cornell (“Using Query Zoning and Correlation Within SMART: TREC 5” by Chris Buckley, Amit Singhal and Mandar Mitra) made three major revisions in their routing runs for TREC-5. The first change was the use of a “query zone” to subset the training documents into 5000 high-ranking non-relevant ones, in addition to the relevant ones, for use in expansion and reweighting (for further work, see [20]). They also used a complex process to expand and reweight the query terms, including a Rocchio method with positive and negative term weighting based more heavily on the training documents rather than on the topic. As a final change, they examined promising co-occurrence terms, with a total of 100 single terms, 10

phrases, and 50 co-occurring word pairs being added to the final query. The DFO algorithm was applied for fine tuning of all weights. These experiments yielded a total of 23% improvement over their TREC-4 algorithms.

INQ303 – University of Massachusetts at Amherst (“INQUERY at TREC-5” by James Allan, Jamie Callan, Bruce Croft, Lisa Bellesteros, John Broglio, Jinxi Xu and Hongmin Shu) used similar techniques to the new algorithms tried in TREC-4 [1]. These consisted of adding up to 250 single terms, adjacent word pairs, and nearby word pairs from the training documents. A complex selection process based on term occurrence in 200-word “best-match” passages was used. These terms and word pairs were then reweighted using the DFO algorithm for optimizing performance.

pircs6 – Queens College, CUNY (“TREC-5 English and Chinese Retrieval Experiments using PIRCS” by K.L. Kwok and L. Grunfeld) implemented a genetic algorithm to select the best training subset as opposed to using only the short or high-ranking subdocuments for training as in TREC-4 [13]. This run is after six generations of “genetic growing”, but produced results only 5% better than using the initial training subset (iteration 0).

Brkly14 – University of California, Berkeley (“Term importance, Boolean conjunct training, negative terms, and foreign language retrieval: probabilistic algorithms for TREC-5” by Fredric C. Gey, Aitao Chen, Jianzhang He, Liangjie Xu and Jason Meggs) used massive automatic query expansion using a chi-square discrimination measure similar to that used in TREC-3 [6]. There were an average of 2032 terms added to the initial query. Some experiments were also run involving maximizing the weighted contributions from the top 15 terms added by including these terms specifically in the final (50) regression equations used in retrieval.

uwgrcr0 – University of Waterloo (“Interactive Substring Retrieval (MultiText Experiments for TREC-5)” by Charles L.A. Clarke and Gordon V. Cormack) used queries that were manually built in a special query language called GCL. The routing experiments involved interactive query construction based on co-occurrence of substrings in the training documents.

³Of the 50 original routing topics, one was mistakenly not judged, four had no relevant documents in the test document set, and six had one or two relevant documents in the test set. The appendix includes the evaluation results over the 45 topics that had at least one relevant document.

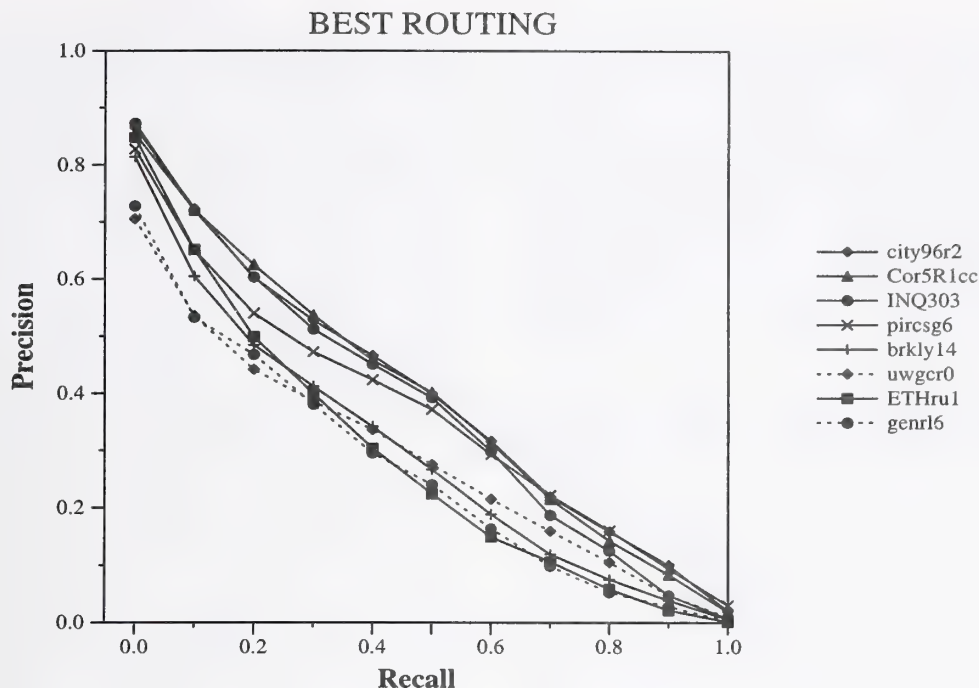


Figure 10: Recall/Precision graph for the top eight routing runs.

ETHru1 – Swiss Federal Institute of Technology (ETH) (“SPIDER Retrieval System at TREC-5” by Jean-Paul Ballerini, Marco Büchel, Ruxandra Domenig, Daniel Knaus, Bojidar Mateev, Elke Mittendorf, Peter Schäuble, Páraic Sheridan, and Martin Wechsler) investigated various feature selection methods, including the method used by OKAPI (RSV), the chi-square method, and the U method. They report that the U method worked consistently the best, likely because it is based on only positive correlations of feature occurrences. In addition to these experiments, they also tried experiments using co-occurring terms within sentences or paragraphs. These were formally motivated by the OKAPI RSV values to create independent indices of the documents, and the results were combined linearly using logistic regression for parameter discovery.

genrl6 – GE / Lockheed Martin / NYU / Rutgers (“Natural Language Information Retrieval: TREC-5 Report” by Tomek Strzalkowski, Louise Guthrie, Jussi Karlgren, Jim Leistensnider, Fang Lin, Jose Perez-Carballo, Troy Straszheim, Jin Wang and Jon Wilding) developed a new stream architecture to investigate combining of results from three different search engines using various types of data as input to those engines. This

group has always concentrated on using more sophisticated natural language processing techniques to locate different types of language structures. These structures are problematic to combine for a single set of results, and the stream architecture provides the flexibility needed to investigate optimal combinations. The *genrl6* run combined four streams of input (stems, collocation, pairs and names) within the PRISE system with results from a new classification-based routing system.

It should be noted that the routing task in TREC has always served two purposes. The first is its intended purpose: to test systems in their abilities to use training data to build effective filters or profiles. The second purpose, which has become equally important in the more recent TRECs, is to serve as a learning environment for more effective retrieval techniques in general. Groups use the relevance judgments to explore the characteristics of relevant documents, such as which features are most effective to use for retrieval or how to best merge results from multiple queries. This is more profitable than simply using the previous TREC results in a retrospective manner because of the use of completely new testing data for evaluation.

A focus on using the training data as a learning environment was particularly prevalent in TREC-5.

Cornell used the relevant and non-relevant documents for investigations of Rocchio feedback algorithms, including more complex processes of expansion and weighting. Waterloo interactively searched the training data for co-occurring substrings and GE (with their partners Lockheed Martin, NYU, and Rutgers) ran major experiments in data fusion to test their new stream-based architecture. In each of these cases the experiments are assumed to lead to better ways of doing the routing task, and also to new approaches for the ad hoc task.

Three experimental themes dominated most routing experiments. The first is the discovery of optimal features (usually single terms) for use in the query or filter. City continued its experiments in repeatedly trying various combinations of terms to discover the optimal set, but for TREC-5 used subsets of the training data. Berkeley concentrated on further investigations of the use of the chi-square discrimination measure to locate large numbers of good terms, and ETH tested three different feature selection methods, including the chi-square method, the RSV (OKAPI) method, and a new method, the U measure. Xerox ("Xerox TREC-5 Site Report: Routing, Filtering, NLP, and the Spanish Tracks" by D. Hull, G. Grefenstette, B. Schulze, E. Gaussier, H. Schütze, and J. Pedersen) also investigated a new feature selection method, the binomial likelihood ratio test.

The second theme was the use of co-occurring term pairs in the training data to "expand" the query. Four groups experimented with locating and incorporating co-occurring pairs of terms, including IN-QUERY in both TREC-4 and TREC-5, and Cornell in TREC-5. As mentioned before, Waterloo interactively looked for word-pairs or co-occurring strings to manually add to their query. ETH used the OKAPI RSV values to formally motivate a series of experiments using co-occurring terms within different portions of the document (within sentence, within paragraph, etc.) as different methods of constructing queries. These multiple representations of the query were then linearly combined, with the parameters for that combination discovered using logistic regression on the training data.

The third theme in the routing experiments was the continuing effort to use only subsets of the training data. The number of judged documents per topic is on the order of 2000 or more, and this can be computationally difficult for complex techniques. Efficiency has motivated CUNY experiments (the PIRCS system) since TREC-3 where they tried using only the "short" documents for training. In TREC-5 this

group used genetic algorithms to select the optimal set of training documents. Cornell (in TREC-5) used a new "query zone" technique to subset the training documents so that not all non-relevant documents were used for training. The goal was not just improved efficiency, but also improved effectiveness in that training was more concentrated on documents that the Cornell system is likely to retrieve.

There is another issue that suggests the use of subsets: the problem of overfitting the queries/methods to the training data. This was specifically emphasized in the City system, where they used different subsets of the training data for locating features, and used combinations of runs for their final results. Xerox used subsets to reduce overfitting, with their subsets based on finding documents within a "local zone" to the query (a predecessor to the query zoning technique used by Cornell). The Xerox paper provides more discussion of the overfitting problem and suggests some additional techniques to avoid it.

As in the ad hoc task, there is a heavy adoption rate across groups for successful techniques. For the ad hoc task these techniques revolve around better ways of handling the initial topic, or use of the top X documents for relevance feedback. Because of the existence of training data in routing, the routing experiments have generally not used the topic itself heavily, but constructed queries mainly based on the training data. The success of these techniques therefore revolves around how well the test data matches the training data, and also on how tuned the techniques are to the particular training data.

Figure 11 shows a comparison of routing results from several of the best performing systems in TRECs 3, 4, and 5. In general, the routing results show little improvement from TREC-3, although most of the systems have clearly developed superior methodologies. This is likely due to poorer matches between the training and test data for TRECs 4 and 5 than for TREC-3. The TREC-3 test data (Disk 3) was extremely similar to the training data, in that Disk 3 had similar content to that of Disks 1 and 2 (see Table 2 for details). In TREC-4 the training data was Disks 1, 2, and 3, but the test data was additional *Federal Register* material for 25 topics, and lots of "net trash" for the other 25 topics. The mixture of two distinctly different types of data, and the use of the very long *Federal Register* documents, made the routing task much more difficult.

TREC-5 used AP documents as training data, with FBIS material for test data. Whereas the types of documents are similar, the domains of the documents do not always match. So for some topics there is a

ROUTING -- TREC-5 VS TREC-4 VS TREC-3

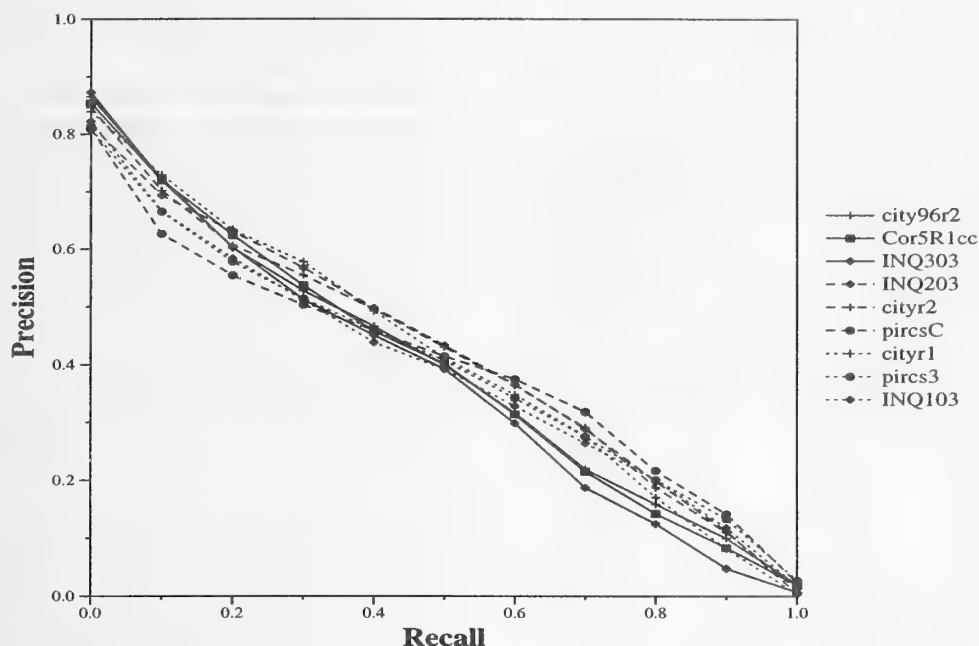


Figure 11: Recall/Precision graph for routing runs from 3 TRECs.

good match of training and test data, but for others the match is very poor, and very few relevant documents were found for those topics. Four topics had zero relevant documents in the test set, and an additional six topics had only one or two relevant documents. Even after dropping the four topics with no relevant documents from the evaluation, the results are still heavily affected by the mismatch.

This effect can be seen in Table 9 where the topic hardness measure (same definition as for the ad hoc results) is given, along with the number of relevant documents in both the test and training sets. As can be seen, there is a definite correlation between the number of relevant documents in the test set and the hardness of the topics (correlation coefficient of 0.812), and considerably less of a correlation (0.497) between the hardness and the number of relevant training documents. The table is sorted by the number of relevant test documents, and the average hardness for topics with ten or fewer relevant documents is 0.085, way below that for topics with many more relevant documents.

Note that this strong correlation between hardness and the number of relevant documents did not occur in the ad hoc task, where there is a very low correlation (less than random) between these factors. Whereas it is difficult to understand the factors that make the ad hoc topics "harder", it is obvious that

one big factor in the "harder" routing topics is the degree of match between the training data and the test data. However, this is a real-world constraint, since in any operational system there will be differences in the degrees of match and systems will need to be able to recognize "mismatches" and adapt for them.

In TREC-6 an attempt will be made to have a close match between the training and test data. This should create a second comparison point for well-matched data with the improved systems, and allow further examination of the effects of the training data issues.

6 Summary

It is difficult to summarize the results of so many groups and experiments. Each group ran multiple experiments that resulted in their TREC submission, and readers are urged to explore the individual papers in this proceedings. Appendix B, "Summary Performance Comparisons TREC-2, TREC-3, TREC-4, TREC-5" by Karen Sparck Jones presents a snapshot of various system performances, particularly in the high precision end of the retrieval spectrum.

However, two main conclusions can be drawn from TREC-5:

- Systems seem to be adjusting to the much

Table 9: Number relevant documents in training and test sets and hardness for TREC-5 routing topics

Topic	Training Relevant	Test Relevant	Hardness
53	389	1	0.0435
207	31	1	0.1304
224	50	1	0.0000
211	190	2	0.1087
222	55	2	0.0000
243	34	2	0.0217
125	183	6	0.1739
23	141	7	0.0994
194	76	8	0.0489
44	57	10	0.0348
192	179	10	0.2783
77	179	14	0.2329
185	105	14	0.1273
173	203	15	0.0957
126	252	18	0.3913
5	40	19	0.2906
154	450	22	0.2945
1	111	30	0.2058
78	200	37	0.3161
24	145	38	0.1968
114	221	42	0.1915
58	144	45	0.3652
54	157	48	0.3333
82	217	55	0.3613
94	119	56	0.1762
123	262	57	0.2853
228	31	68	0.1988
240	168	88	0.1354
11	231	92	0.2864
95	92	93	0.1580
3	74	101	0.3852
161	133	153	0.7061
100	107	157	0.5361
6	129	158	0.3891
108	266	174	0.2839
4	41	178	0.5400
119	461	185	0.3052
221	84	193	0.2687
187	147	194	0.3483
12	272	228	0.4922
118	559	324	0.5109
202	99	583	0.5987
189	882	584	0.6135
142	847	808	0.9126
111	235	887	0.7996

shorter topics in the ad hoc task. Most of the automatic expansion methods tried in TREC-4 were used again in TREC-5, but with significant adjustments to handle the short topic. These adjustments tended to center around getting more information from the topic itself, rather than just extracting keywords. However, comparison runs using the full or long topic still produced over 20% improvement in performance in most cases. The ad hoc runs using manually-built queries (mostly) involved interactivity, since the query construction rules changed in TREC-5 to allow this. Groups either tested human-computer “teamwork”, or involved users in order to better learn how to automatically build ad hoc queries.

- The routing results for TREC-5 were somewhat disappointing. Whereas there were many groups with significant improvements in performance, the overall results were not better than for TREC-4 (or for TREC-3). The problem appears to be the serious mismatch between the training and the test data, which unexpectedly has happened in both TREC-4 and TREC-5. In TREC-5 there was a domain mismatch for many of the topics, resulting in very few relevant documents. While this is not an unrealistic problem — and systems must learn to adapt to it — providing a better match between the training and test data in the future will enable a better evaluation of the new routing methods.

The six TREC-5 tracks significantly expanded the amount of research performed in TREC. Many of the tracks further explored the work initiated in the preliminary running of the track in TREC-4.

Interactive: This was the second running of the interactive track. Based on the lessons learned from the TREC-4 track on how difficult it is to fairly compare results in interactive experiments, the track concentrated on experimental design in TREC-5. Unfortunately, the final design was not decided until late in the TREC cycle, and only two groups were able to participate. The track will continue in TREC-6 using the experimental design developed in TREC-5.

Database merging: This was also the second running of the database merging track, with three groups participating in the track in TREC-5. The track has proved to be a high-overhead track (this year’s task required creating 98 separate databases), and thus has not attracted much participation despite general interest in the prob-

lem. The track will likely be run again in future TRECs, but will not be run in TREC-6.

Multilingual: Seven groups submitted Spanish runs and nine groups submitted Chinese runs. As in TREC-4, the Spanish results demonstrated that many of the techniques used in English retrieval can be successfully applied to Spanish. Given the success of traditional techniques on Spanish, it was decided to discontinue the Spanish portion of the multilingual track.

This was the first year for Chinese in TREC, and most groups concentrated on segmentation issues. The Chinese track will continue in TREC-6.

Confusion: A confusion (or data corruption) track was run in TREC-4 in which characters were randomly changed to simulate the type of output one might get from an Optical Character Recognition (OCR) process. In TREC-5, the test data was actual OCR output of scanned images of the 1994 *Federal Register*. Five groups participated in the experiment designed to explore the effect different levels of OCR error has on retrieval performance.

The track introduced the known-item search as a new task for TREC. The known-item search task will be used again in the Spoken Document Retrieval (SDR) track, a new track to begin in TREC-6. The SDR track is a successor to the confusion track in that it represents a different form of "corrupted" documents. Instead of retrieving documents that are the result of OCR, systems will retrieve documents that are the result of speech recognition systems. The interaction between OCR and retrieval will continue to be explored in the new METTREC workshop.

Filtering: The TREC-5 filtering track followed the same design as the preliminary track in TREC-4, and had seven participating groups. The goal in the track was to retrieve an unranked set of documents that optimizes a pre-specified utility function. A family of three functions was used to investigate how retrieval was affected by changes in the relative worth of retrieving a relevant document versus not retrieving a nonrelevant document. The track will continue into TREC-6 with a different set of utility functions and a set of test documents that more closely matches the training data.

NLP: Four groups participated in the initial running

of the natural language processing track. The track will continue in TREC-6.

In addition to the tracks above, TREC-5 had a "trial" run of the Very Large Corpus (VLC) track. The VLC track will have its first official running in TREC-6, where participants will perform ad hoc searches on approximately 20 gigabytes of text. Other new tracks in TREC-6 will be a Cross Language track, in which systems use topics in one language to retrieve documents in a second language, and a High-Precision track, where participants attempt to retrieve the best 10 documents for each topic using no more than five minutes (wall clock time) for each topic.

Acknowledgments

The authors would like to gratefully acknowledge the continued support of the TREC conferences by the Intelligent Systems Office of the Defense Advanced Research Projects Agency. Special thanks also go to the TREC program committee and the staff at NIST.

References

- [1] J. Allan, L. Ballesteros, J. Callan, B. Croft, and Z. Lu. Recent experiments with INQUERY. In *Proceedings of the Fourth Text REtrieval Conference (TREC-4)*, pp. 49–63, 1996. NIST Special Publication 500-236.
- [2] Eric Brown. Fast evaluation of structured queries for information retrieval. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 30–38, 1995.
- [3] C. Buckley and G. Salton. Optimization of relevance feedback weights. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 351–357, 1995.
- [4] Robert Burgin. Variations in relevance judgments and the evaluation of retrieval performance. *Information Processing and Management*, 28(5):619–627, 1992.
- [5] W. Cooper, A. Chen, and F. Gey. Full text retrieval based on probabilistic equations with coefficients fitted by logistic regression. In *The Second Text REtrieval Conference (TREC-2)*, pp. 57–66, 1994. NIST Special Publication 500-215.

- [6] W. Cooper, A. Chen, and F. Gey. Experiments in the probabilistic retrieval of full text documents. In *Overview of the Third Text REtrieval Conference (TREC-3) [Proceedings of TREC-3]*, pp. 127–134, 1995. NIST Special Publication 500-225.
- [7] Larry Fitzpatrick and Mei Dent. Automatic feedback using past queries: Social searching? In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR-97)*, pp. 306–313, 1997.
- [8] Donna Harman. Analysis of data from the second Text REtrieval Conference (TREC-2). In *Proceedings of RIAO94*, pp. 699–709, 1994.
- [9] Donna Harman. Overview of the fourth Text REtrieval Conference (TREC-4). In D. K. Harman, editor, *Proceedings of the Fourth Text REtrieval Conference (TREC-4)*, pp. 1–23, October 1996. NIST Special Publication 500-236.
- [10] Stephen P. Harter. Variations in relevance assessments and the measurement of retrieval effectiveness. *Journal of the American Society for Information Science*, 47(1):37–49, 1996.
- [11] D. Hawking and P. Thistlewaite. Searching for meaning with the help of a PADRE. In *Overview of the Third Text REtrieval Conference (TREC-3) [Proceedings of TREC-3]*, pp. 257–267, 1995. NIST Special Publication 500-225.
- [12] K.L. Kwok. A new method of weighting query terms. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 187–196, 1996.
- [13] K.L. Kwok and L. Grunfeld. TREC-4 ad-hoc, routing retrieval and filtering experiments using PIRCS. In *Proceedings of the Fourth Text REtrieval Conference (TREC-4)*, pp. 145–152, 1996. NIST Special Publication 500-236.
- [14] M. E. Lesk and G. Salton. Relevance assessments and retrieval system evaluation. *Information Storage and Retrieval*, 4:343–359, 1969.
- [15] D. Evans N. Milić-Frayling and R. Lefferts. CLARIT TREC-4 experiments. In *Proceedings of the Fourth Text REtrieval Conference (TREC-4)*, pp. 305–321, 1996. NIST Special Publication 500-236.
- [16] A. Moffat and J. Zobel. Information systems for large document collections. In *Overview of the Third Text REtrieval Conference (TREC-3) [Proceedings of TREC-3]*, pp. 85–93, 1995. NIST Special Publication 500-225.
- [17] S.E. Robertson, S. Walker, and M.M. Hancock-Beaulieu. Large test collection experiments on an operational, interactive system: Okapi at TREC. *Information Processing and Management*, 31(3):345–360, 1995.
- [18] S.E. Robertson, S. Walker, S. Jones, M.M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In *Overview of the Third Text REtrieval Conference (TREC-3) [Proceedings of TREC-3]*, pp. 109–126, 1995. NIST Special Publication 500-225.
- [19] A. Singhal, C. Buckley, and M. Mitra. Pivoted document length normalization. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 21–29, 1996.
- [20] A. Singhal, M. Mitra, and C. Buckley. Learning routing queries in a query zone. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 25–33, 1997.
- [21] K. Sparck Jones. Reflections on TREC. *Information Processing and Management*, 31(3):291–314, 1995.
- [22] K. Sparck Jones and C. van Rijsbergen. Report on the need for and provision of an “ideal” information retrieval test collection. British Library Research and Development Report 5266, Computer Laboratory, University of Cambridge, 1975.
- [23] Jean Tague-Sutcliffe and James Blustein. A statistical analysis of the TREC-3 data. In D. K. Harman, editor, *Overview of the Third Text REtrieval Conference (TREC-3) [Proceedings of TREC-3]*, pp. 385–398, April 1995. NIST Special Publication 500-225.
- [24] J. Xu and W.B. Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 4–11, 1996.

TREC-5 Interactive Track Report

Paul Over

National Institute of Standards, Gaithersburg, MD 20899

1 Outline

This report presents the framework for the TREC-5 interactive track experiments in detail, cites the results but refers the reader to the site reports for their analysis, and discusses unexpected disagreements between relevance and aspectual assessment performed by the NIST assessors.

- Goals
- Experimental Design
- Execution and Results
- Review of Assessor Disagreements
- References
- Appendix A - Task specification
- Appendix B - General instructions and topic text
- Appendix C - Reporting results to NIST
- Appendix D - Instructions for NIST aspect assessors
- Appendix E - Summary evaluation
- Appendix F - Supporting material for consistency review

2 Goals

The high-level goal of the Interactive Track in TREC-5 was the investigation of searching as an interactive task by examining the process as well as the outcome.

As more people use or procure interactive search systems the ability to understand their relative utility within a given context becomes more important. In order to build better interactive systems much more needs to be known about how humans interact with such systems in filling their various needs for information. To these ends an experiment was designed, including:

1. a task for the searchers participating in the experiment
2. agreements on data recording and formats for results reporting
3. measures

3 Experimental Design

3.1 Task

A minimum of four searchers per site were each to perform six searches on the TREC-5 adhoc collection, drawing topics from a set of 12 especially chosen adhoc topics modified for use in the interactive track. Three of the searches were to be on the site's experimental system and three on a control system, the purpose of which was to aid in comparing systems across sites. See Appendix A for details on the experimental design.

The topics described an information need with many aspects - an aspect being roughly one of many possible answers to a question which the topic in effect posed. For example, topic 254 asks the question "What kinds of medications or procedures other

than heart surgery have been used to treat heart ailments ?” Searchers found aspects/answers such as angioplasty, lasers, stents, etc. in the documents of the collection. Language was added to each topic to make the meaning of aspect for that topic clearer. See Appendix B for the general instructions to the searchers and the text of the topics.

The task of the interactive searchers was to save documents, which, taken together, covered as many different aspects of the topic as possible in the 20 minutes allowed per search. The number of documents saved was not important. Participants were not penalized for saving documents that covered a given aspect more than once.

The task posed special challenges to participating systems and searchers. Beyond the usual goal of finding documents about a particular topic, it would seem to favor systems which also support the searcher in efficiently:

1. isolating and comparing material in a document about different aspects of the topic: “the document pertains to the topic but which, if any, specific aspects are covered and where?”
2. avoiding other documents with mostly the same or similar material: “once I find one or more aspects, how much help, if any, do I get in avoiding documents and document sections which cover already identified aspects?”

Two sorts of result data were to be collected. Sparse format data for each search comprised the list of documents saved and the elapsed time of the search. Rich format data recorded significant events in the course of the interaction and their timing. Sparse format data were to be the basis for the summary evaluation, which produced a triple for each search: precision, aspectual recall, and time. Rich format data were intended for analytical evaluation by the experimenters. See Appendix C for information on the format of the results reported to NIST.

3.2 Interactive Track Assessment and Evaluation at NIST

For each topic, a pool was formed containing the unique documents saved by at least one searcher for that topic regardless of site. The measures for the interactive track required two sorts of assessments: traditional relevance assessment and a new aspectual assessment.

Relevance assessment for the interactive track amounted only to judging those few documents not already judged for the main TREC task. Every document in the pool was assessed for aspects regardless of whether it had been judged relevant or not.

Aspectual assessment required new procedures. Where possible (there were two exceptions) the assessor who had performed the relevance assessment was assigned to do the aspectual analysis. For each topic, the aspect assessor was asked to:

1. read the topic carefully
2. read each of the documents to be judged and gradually:
 - (a) create a list of the aspects found somewhere in the documents
 - (b) select and record a short phrase describing each aspect found
 - (c) determine which documents contain which aspects
 - (d) bracket each aspect in the text of the document in which it was found (as this assessment task was new, the assessors worked with paper and pencil)

The NIST assessors were given the same topic information as the interactive track searchers except that the paragraph that begins “Please save at least one document” was systematically replaced with a section called “Answers”, for example in the case of topic 256i:

Please save at least one document that identifies EACH DIFFERENT criticism of this trend. If one document discusses several criticisms, then you need not save other

documents that repeat those aspects, since your goal is to identify the different criticisms that have been made.

was be replaced by

Answers: each different criticism of this trend.

In dealing with the assessors, the term “aspect” was replaced with the term “answer.” The assessors seemed to have little difficulty understanding the basic task. The questions that did arise concerned mainly the granularity of the answers/aspects, e.g., “should I lump these 3 low-level answers together and count them as 1?” See Appendix D for details on the instructions given to the NIST interactive track assessors.

3.3 Summary Evaluation

For each search (by a given participant for a given topic at a given site), NIST:

1. used the submitted list of selected documents and the assessor’s aspect-document mapping for the topic to calculate the percentage of aspects covered by the submitted documents.
2. calculated the precision for the submitted document set using the standard relevance judgments
3. took the elapsed clock time for the search from the submitted data

3.4 Analytical Evaluation

No analytical evaluation was performed at NIST. Additional information may be available in the site reports from Rutgers University and City University, London in the TREC-6 Proceedings.

4 Execution and Results

A relatively late start, problems installing the control system and indexing the collection for its use, and in

one case problems setting up the experimental system resulted in the two participating systems’ not being able to execute the complete experimental design. No searches using the control system were run. Searches assigned to the control system by the design were instead performed on the experimental system.

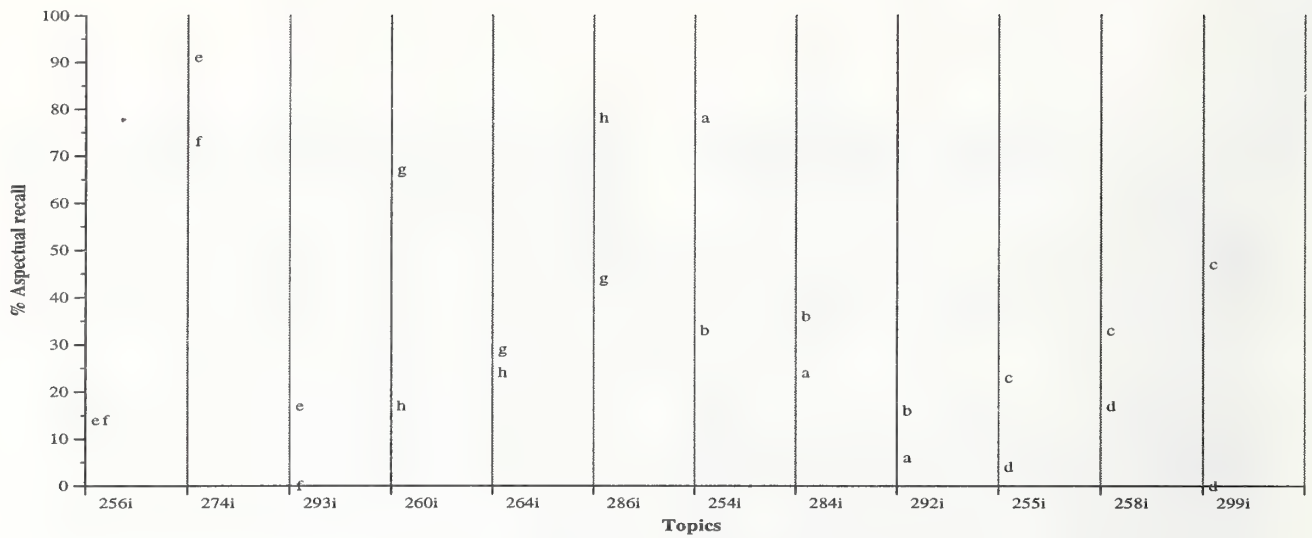
The summary measures for each system are presented separately in Appendix E. The two sites’ results are not directly comparable to each other, not only because no control was used, but also because the collections searched were different. See the site reports from City University, London and Rutgers University in the TREC-5 Proceedings for more information about the actual execution of experimental design.

Even if the experiments as run do not allow comparison of systems, exploratory analysis of the results for each system underlines the importance of dealing in any experimental design with some or all of the following:

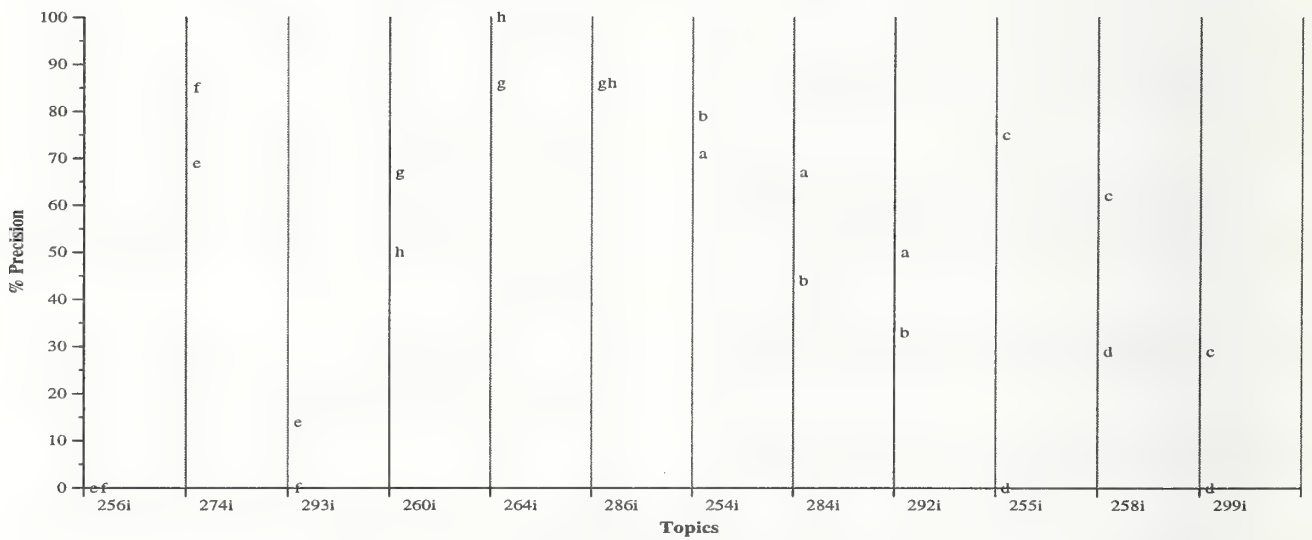
- a strong searcher effect - within a single system and for a single topic, results vary greatly for different searchers
- a strong topic effect - within a single system and for a single searcher, results vary greatly for different topics
- a searcher-topic interaction - within a single system, the effect of different searchers on the results varies for different topics

Figure 1 and Figure 2 depict the 2 main measures for each system by topic and with searcher information encoded in the data points. For example in the upper graph for the results from Rutgers the second column from the left shows the results for topic 274i. Participants “d” and “m” each completed a search and saved documents which covered about 65% of the total number of aspects identified by the NIST assessors.

See for the site reports from City University, London and Rutgers University for more information about the actual execution of experimental design and discussions of an analytical nature.

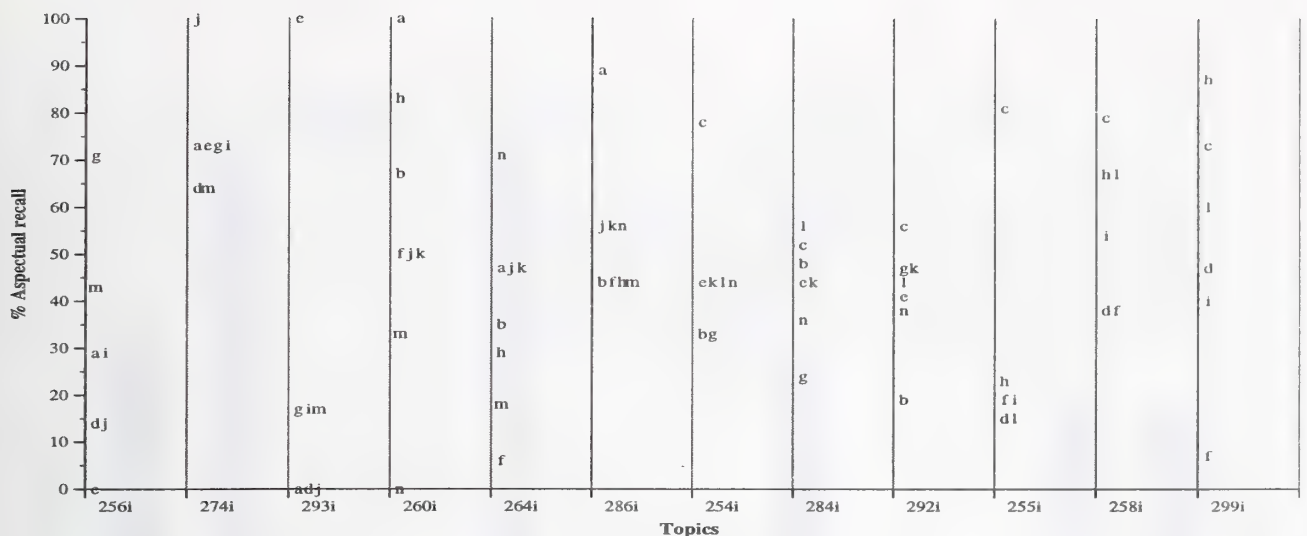


Aspectual recall for all searches by topic
Searchers: a=and b=kar c=bri d=col e=cla f=lis g=ala h=sar



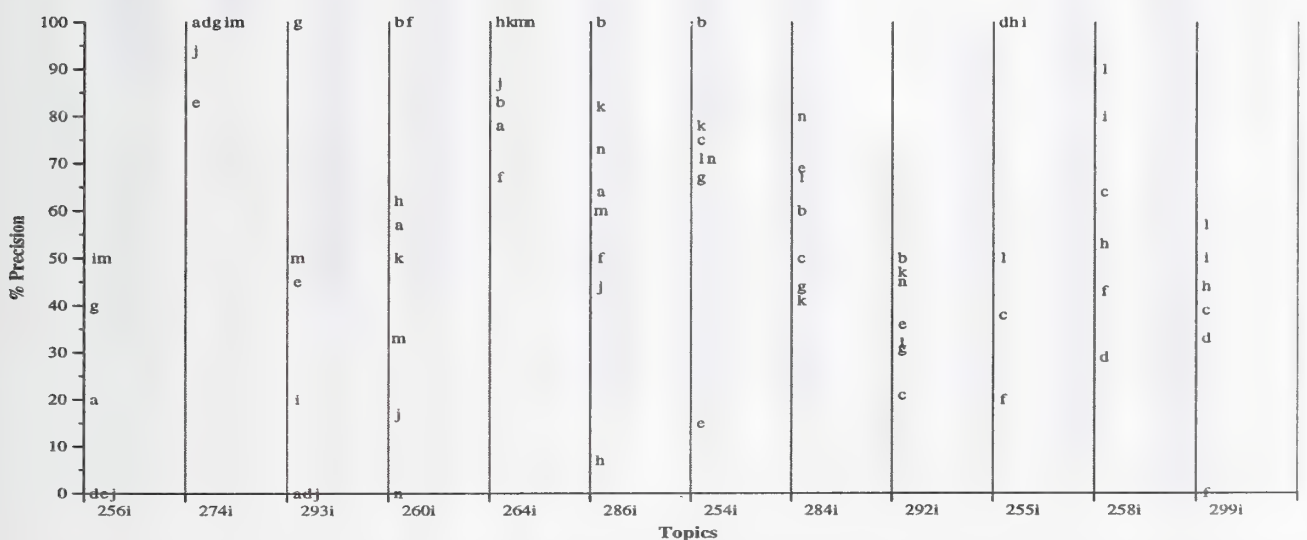
Precision for all searches by topic
Searchers: a=and b=kar c=bri d=col e=cla f=lis g=ala h=sar

Figure 1: Results from City University, London.



Aspectual recall of all searches by topic

Searchers: a=s001 b=s002 c=s003 d=s004 e=s005 f=s006 g=s007 h=s008 i=s009 j=s010 k=s011 l=s012 m=s013 n=s014



Precision of all searches by topic

Searchers: a=s001 b=s002 c=s003 d=s004 e=s005 f=s006 g=s007 h=s008 i=s009 j=s010 k=s011 l=s012 m=s013 n=s014

Figure 2: Results from Rutgers University.

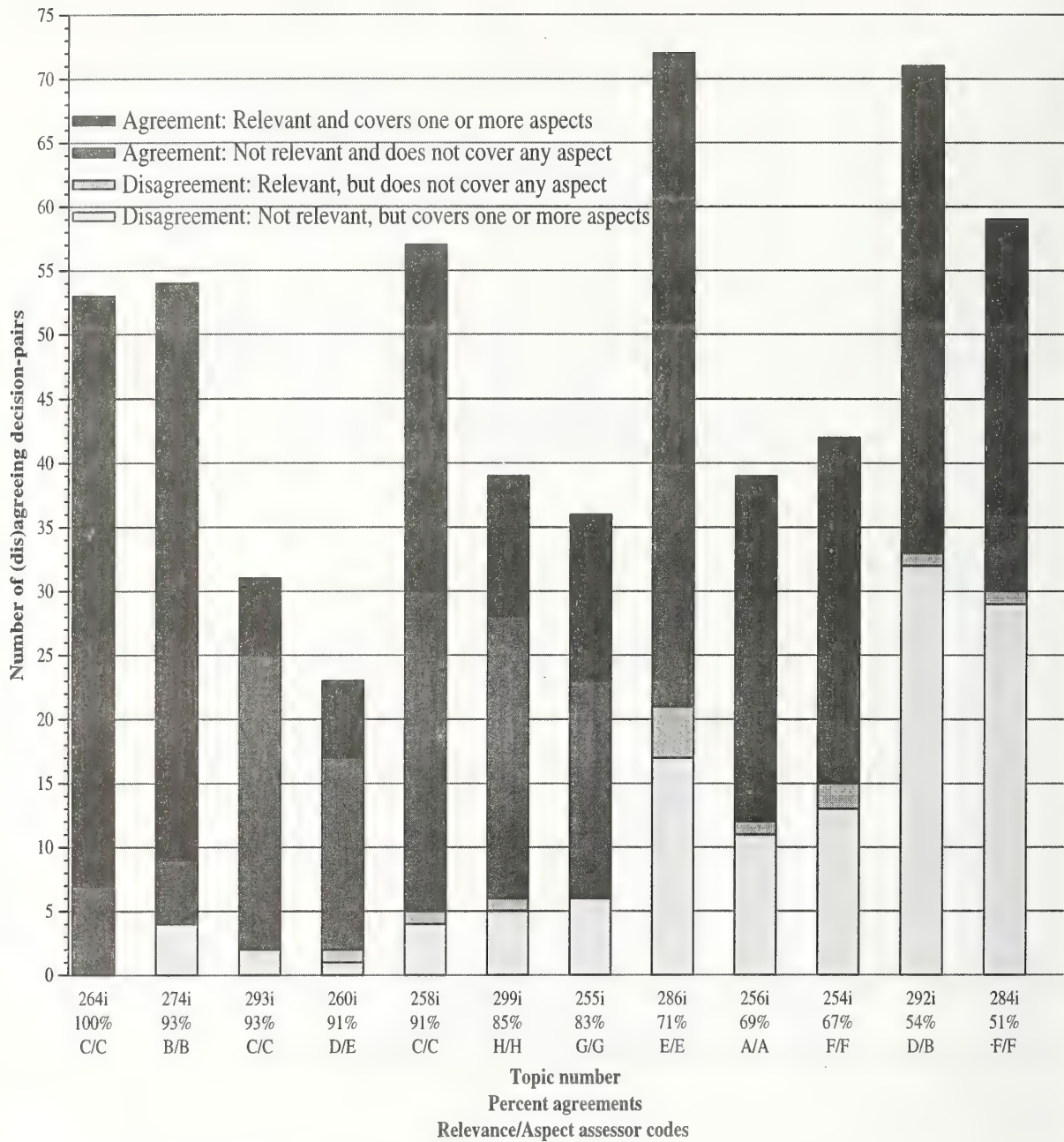


Figure 3: NIST Assessor (dis)agreements in relevance versus aspect judgements by topic.

5 Review of Assessor Disagreements

In the process of evaluating the results submitted to NIST, the assessors answered two questions for each of the 583 documents in two sessions within 2 weeks of each other.

- “Is it relevant to the topic?”
- “What, if any, aspects does it cover?”

For 10 of the 12 topics, the relevance assessor was the same person as the aspect assessor. For two topics, the relevance and aspect assessors were not the same person.

The designers of the interactive task assumed there would be a high degree of consistency between relevance and aspect assessment results. Being relevant was expected to imply covering one or more aspects and vice versa, but the TREC-5 interactive track assessments appear to violate this assumption in a significant number of cases. As part of the follow-up to TREC-5 conference, the author reviewed the TREC-5 interactive track data and relevant, available literature on consistency of relevance judgements in an attempt to answer the following initial questions:

1. How much inconsistency exists in the TREC-5 interactive track assessments and of what type(s) is it?
2. Are the levels of consistency or lack of it within “normal” bounds?
3. For any abnormal levels, are there any likely explanations?

5.1 Amount and Types of Inconsistency

5.1.1 The Data

Figure 4 shows the number of judgements by type: total agreements, total disagreements, and then the two possible types of disagreements:

1. “Document is relevant, but does not cover any aspects”

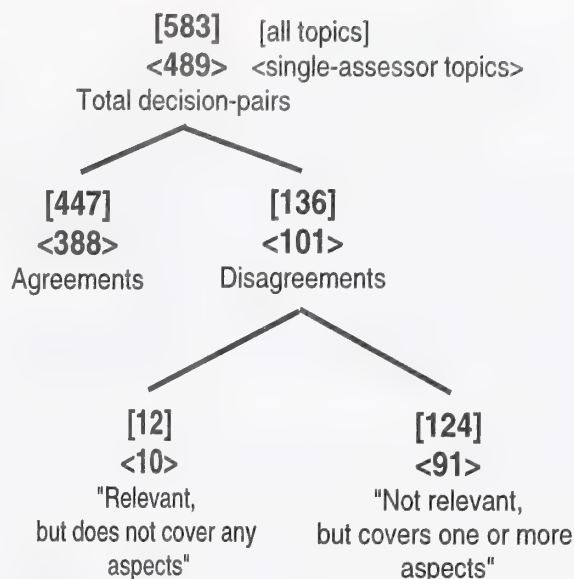


Figure 4: Categorization of assessments.

2. “Document is not relevant, but does cover one or more aspects”

Looking at all topics, relevance and aspect assessments are in agreement 77% of the time (80% of the time in the case of the single-assessor topics). The disagreements are unequally distributed with about 90% being of the type “Document is not relevant, but does cover one or more aspects” - another unexpected result.

Figure 3 shows more detail, for each topic: the number of assessor agreements and disagreements by type, the percentage of agreements, and the relevance and aspect assessors. The topics are arranged from left to right in order of decreasing percentage of agreements. For example, the sixth column from the left represents the data on topic 299. Assessor H made both the relevance and the aspectual coverage assessments, which agreed 85% of the time, i.e., for 33 of 39 documents the judgements were: “Relevant and covers one or more aspects” or “Not relevant and covers no aspect.” There was one document judged to be relevant but not to cover any aspect, and there were five documents judged to be not relevant, but found

to cover one or more aspects.

5.1.2 Two Errors in Results Processing

In the course of reviewing the documents on which NIST assessors disagreed, two errors made in the assessment recording were uncovered. The information presented here on the review of consistency reflects the corrected results, but because the effect would be small and in order to maintain consistency with the data on which the site reports were based, the summary evaluation presented here has not been corrected.

For topic 254 the document WSJ920225-0057 was assessed as covering aspect 1 but did not. The relevance assessment as "not relevant" stands. The aspectual recall of the Rutgers University search 254i-005-2 might have been affected, but was not because another document saved by the searcher also covered aspect 1. For topic 284 the document AP880310-0205 was incorrectly recorded as covering aspect 1. The relevance assessment as "not relevant" stands. Correcting the error would reduce by 4% the aspectual recall of the following searches:

- City University 284_and
- City University 284_kar
- Rutgers University 284i-003-1
- Rutgers University 284i-011-1

5.2 What's normal consistency?

Are the levels of consistency or lack of it within "normal" parameters? The published literature on relevance assessments contains mostly studies which measure differences between different individuals' assessments of the same material, not between the same individual's. The heterogeneity of the between-assessor studies with regard to assessor characteristics, type/length of material judged, type of judgement, similarity measure, etc., makes comparison very difficult, but, with that caveat, one might minimally expect the TREC-5 within-assessor data to exhibit greater consistency than the between-assessor data, and for the most part the TREC-5 data do.

By any of the measures in Figure 5, averaged over all the single-assessor topics, the TREC-5 relevance and aspect judgements agree to a greater extent than most of those in the published studies comparing different individuals' assessments of the same topic-document pairs. (Note: As can be seen from Figure 5 the 2 two-assessor topics have little effect on the average across all topics but to simplify comparison with other work, references here will be to the single-assessor topics only, unless explicitly noted.)

For example, with regard to the ability of one judge to duplicate the judgement of another as measured by "effectiveness" (where 0 is no agreement and 1 is perfect agreement) (Goffman & Newill, 1966) the following averages have been reported or can be calculated from reported data:

- 0.35 (Barhydt, 1967)
- 0.30 (Janes & McKinney, 1992)
- 0.495 (Janes, 1994)
- 0.58 TREC-5 Interactive

Similarly, with regard to agreement on relevant documents as defined by Lesk and Salton (1969):

- 0.31 (Lesk & Salton, 1969)
- 0.52 (Janes & McKinney, 1992)
- 0.36 - 0.64 (Burgin, 1992)
- 0.70 TREC-5 Interactive

With regard to direct comparisons:

- 57.2% (Figueiredo, 1978) as reported by Janes (1994)
- 66% and 70% (Janes & McKinney, 1992)
- 80% TREC-5 Interactive

With regard to correlation:

- 0.55 - 0.75 for subject experts (Saracevic, 1975)
- 0.45 - 0.60 for "information providers" (Saracevic, 1975)

Topic Number		Decision-pair types: Relevance - Aspects R = Relevant(Covers aspect) N = Not Relevant(No aspect)				% Decision Pairs that agree 100* (RR+NN)/ All	Measures of Agreement						
		RR	RN	NR	NN		Pearson's r	Agreement on Relevant Docs RR / (RR+NR)	Agreement on Non-Relevant Docs NN / (NN+NR)	Effectiveness of Aspect Assessment in Approximating Relevance Assessment			
										Sensitivity RR / (RR+RN)	Specificity NN / (NN+NR)	Effectiveness Se+Sp-1	
	264	53	0	0	7	100	1.00	1.00	1.00	1.00	1.00	1.00	
	274	45	0	4	5	93	0.71	0.92	0.56	1.00	0.56	0.56	
	293	6	0	2	23	93	0.83	0.75	0.92	1.00	0.92	0.92	
	260	6	1	1	15	91	0.79	0.75	0.86	0.86	0.94	0.79	
	258	27	1	4	25	91	0.83	0.84	0.83	0.96	0.86	0.83	
	299	11	1	5	22	85	0.69	0.65	0.79	0.92	0.81	0.73	
	255	13	0	6	17	83	0.71	0.68	0.74	1.00	0.74	0.74	
	286	32	4	17	19	71	0.45	0.60	0.47	0.89	0.53	0.42	
	256	5	1	11	22	69	0.37	0.29	0.65	0.83	0.67	0.50	
	254	22	2	13	5	67	0.31	0.59	0.25	0.92	0.26	0.19	
	292	15	1	32	23	54	0.31	0.31	0.41	0.94	0.42	0.36	
	284	23	1	29	6	51	0.27	0.43	0.17	0.96	0.17	0.13	
All topics	Totals	258	12	124	189	Avg.	77	0.58	0.65	0.58	0.95	0.60	0.55
1-assessor topics		237	10	91	151	(weighted)	80	0.61	0.70	0.60	0.96	0.62	0.58

Figure 5: Various measures of agreement between the TREC-5 relevance and aspect assessments.

TREC conference	Assessors per document	Average percent of agreement	Type of agreement	Number of documents	Judgment type(s) per document
TREC-4	3	81%	2-way	14968	relevance relevance relevance
TREC-5	1	80%	2-way	489	relevance aspect
TREC-2	2	79%	2-way	6867	relevance relevance
TREC-5	1 or 2	77%	2-way	583	relevance aspect

Figure 6: Summary of TREC assessment consistency data.

• 0.61 TREC-5 Interactive

Exceptions include Shaw, Wood, Wood, and Tibbo (1991), in which direct comparison shows agreement of 98% or better by experts who were allowed to confer and resolve disagreements; Cleverdon (1970), who cites rank correlations "never below 0.92" on full Cranfield documents; the direct comparison, which can be calculated from raw data reported by Lesk and Salton: 98.4% overall agreement on abstracts versus TREC-5's average for single-assessor topics of 70%; and Rees (1967) who reports a correlation (r) of 0.72 on abstracts by judges with varying degrees of medical knowledge.

An example of within-assessor consistency in relevance judgements can be found in a follow-on study performed by Rees et al. (1967). In a follow-up to their larger study Rees et al. had 21 medical students rate abstracts of 12 documents with regard to 5 "aspects" on 11-point scales in two sessions 2 weeks apart. The average correlation per student calculated from the published raw data on the first "aspect" (overall relevance) is 0.72. For the TREC-5 data on relevance/aspectual coverage of full documents, the average correlation for single-assessor topics is 0.61.

TREC-5's within-assessor agreement of 80% lies surprisingly close to the between-assessor agreement for TREC-2 (79%) and TREC-4 (81% for decision pairs). During the assessment of TREC-2 results, 6867 documents were judged for relevance by a second assessor. For TREC-4 14,968 documents were judged for relevance by a second and a third assessor. Figure 6 summarizes the existing TREC data on consistency of assessments. One might have expected greater agreement where only one assessor was involved but less to the extent that the tasks - judging relevance versus judging aspectual content - were perceived as different.

5.3 Likely Explanations?

In an informal attempt to understand some of the variation in consistency by topic, the author read all of the documents on which the TREC-5 interactive track assessors disagreed and reviewed the relevance and aspect assessments. The author disagreed

to varying degrees both with the assessors' relevance judgements and with their aspectual decisions, but no pattern was detectable and so no results of this examination are reported except for the two topics showing the highest levels of disagreement: 284 and 292. The following were examined for these two topics:

1. topic text
2. documents judged against the topic for TREC-5 interactive
3. assessor assignments
4. relevance assessor comments
5. aspect assessor comments

Topic 284 - International drug enforcement cooperation Of 29 disagreements all are of the type "Not relevant, but covers aspects." At the risk of just adding one more set of judgements, it is very difficult after reviewing the documents to understand how 26 of them could have been judged as "not relevant" unless different interpretations of the topic were used in relevance and aspect assessment. However, the relevance assessor also performed the aspect assessment. The assessor provided comments after making the relevance judgements and later after making the aspectual assessments. The comments provide no evidence of any change in interpretation between relevance assessment and aspect assessment. See Appendix F for assessor comments and extracts from documents, which would seem to make them relevant.

Topic 292 - Worldwide welfare The topic description deals in rather fuzzy terms e.g., "social programs" and "poor people." The narrative lists a number of characteristics a relevant document "should" or "would" have - raising the possibility that judgements could differ due to different assumptions about the degree to which each of these characteristics (at least the "would's") is required.

Of 33 disagreements all but one were of the type "Not relevant, but covers aspects" The aspect assessor bracketed each aspect identified in the document being judged. Few if any of the bracketed

answers/aspects meet the majority of the criteria in the narrative, much less all.

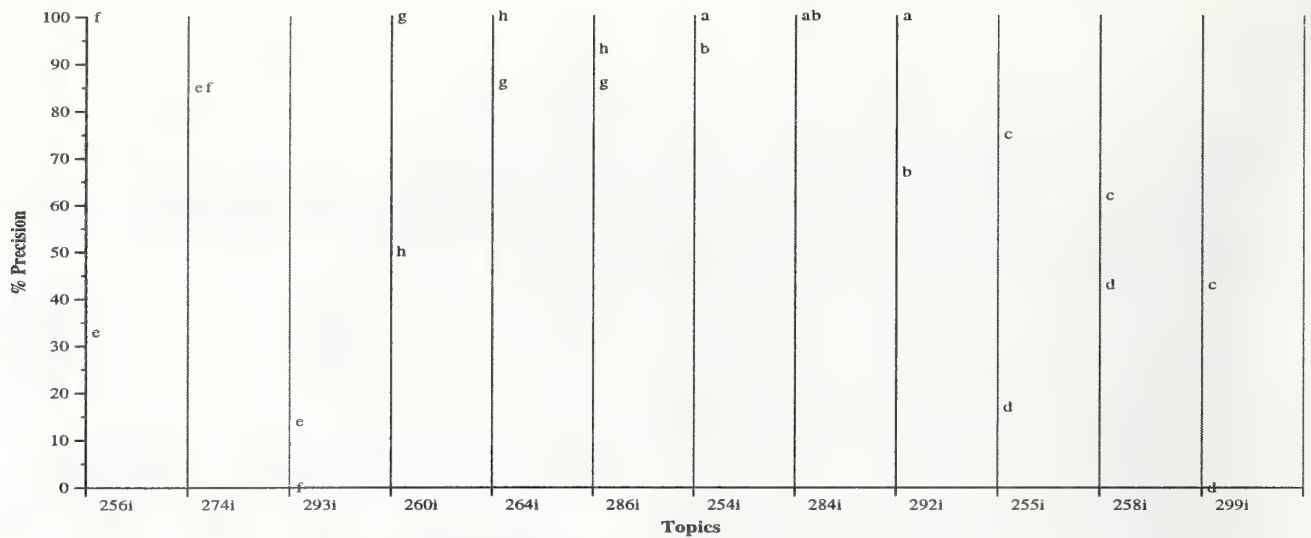
The relevance assessor, who was different from the aspect assessor, made comments which indicate the criteria in the topic narrative were taken as a list of requirements, although exceptions were made. In contrast, the aspect assessor's comments indicate no such strict interpretation and the bracketed aspects suggest a consistently looser view of the topic narrative. It should be noted that the other topic (260) for which aspect assessment was done by a different person than did relevance assessment also has a narrative which lists several criteria for relevance, but this topic had a high level of between-assessor agreement.

5.4 Summary with respect to assessor disagreements

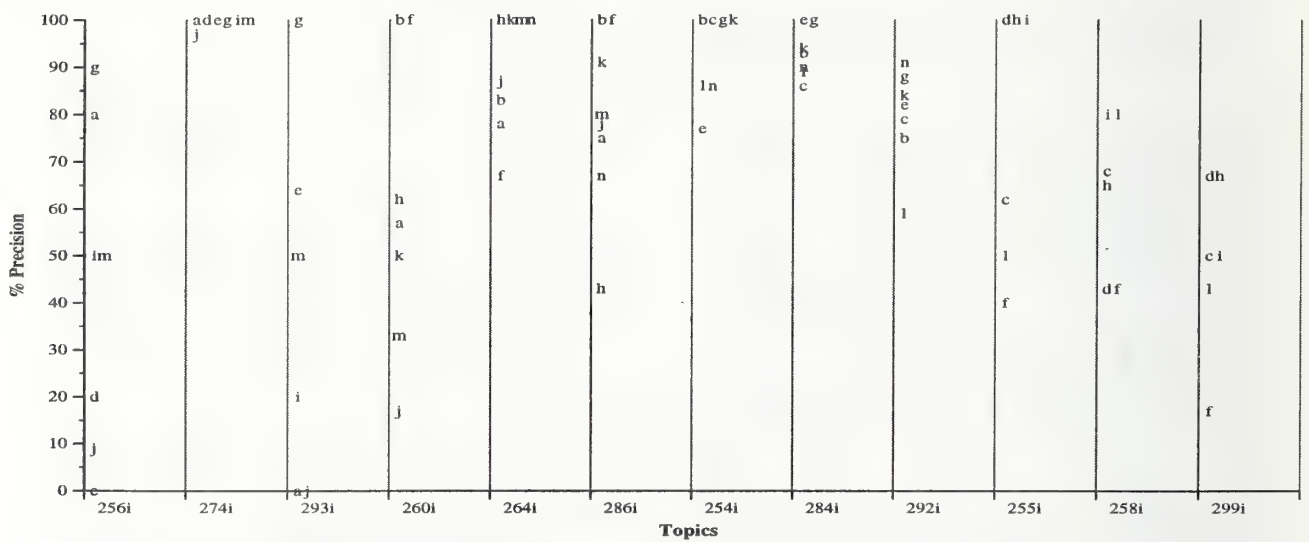
The levels of agreement between relevance and aspect judgements for the TREC-5 interactive assessments are in rough agreement with the reported data for repeated assessments by a single assessor. The notion that the lack of even greater agreement is due largely to differences in how the assessors' perceived the two tasks - relevance and aspect assessment - requires further testing in a better designed, controlled setting. The fact that disagreements were overwhelmingly of the sort "Not relevant, but does cover aspects" seems like additional evidence that the concepts of being relevant and covering aspects were understood as different and that relevance taken more narrowly than aspectual coverage, but there is no independent evidence for this assumption. Neither can we move beyond speculation as to why topics 284 and 292 generated the most disagreement. Finally, if the TREC-5 interactive task specification is reused in TREC-6, it would seem advisable to define precision in terms of the aspect assessment rather than in terms of relevance assessment. Figure 7 shows the results that would have been obtained if precision had been based on aspect assessment rather than the standard relevance assessments.

References

- Barhydt, G. C. (1967). The effectiveness of non-user relevance assessments. *The Library Quarterly*, 23(2), 146-149.
- Burgin, R. (1992). Variations in relevance judgements and the evaluation of retrieval performance. *Information Processing and Management*, 28(5), 619-627.
- Cleverdon, C. W. (1970). *The effect of variations in relevance assessments in comparative experimental tests of index languages* (Cranfield Library Report No. 3). Cranfield, UK: Cranfield Institute of Technology.
- Figueiredo, R. C. (1978). [Estudo comparativo de julgamentos de relevância do usuário e não-usuário de serviços de D. S. I.]. *Ciência da Informacao*, 7, 69-78.
- Goffman, W. and Newill, V. A. (1966). A methodology for test and evaluation of information retrieval systems. *Information Storage and Retrieval*, 3, 19-25.
- Janes, J. W. (1994). Other people's judgements: a comparison of users' and others' judgements of document relevance, topicality, and utility. *Journal of the American Society for Information Science*, 45(3), 160-171.
- Janes, J. W. and McKinney, R. (1992). Relevance judgements of actual users and secondary judges: a comparative study. *The Library Quarterly*, 62(2), 150-168.
- Lesk, M. E. and Salton, G. (1969). Relevance assessment and retrieval system evaluation. *Information Storage and Retrieval*, 4, 343-359.
- Rees, A. M. (1967). Evaluation of information systems and services. In C. A. Cuadra (Ed.), *Annual review of information science and technology* (Vol. 2, pp. 343-359). John Wiley & Sons.
- Rees, A. M., Schultz, D. G., Baumanis, G., Marcus, S., Rothenberg, L., Saracevic, T., Stern,



Precision based on aspect assessments for all searches by topic - City University, London
Searchers: a=and b=kar c=bri d=col e=cla f=lis g=ala h=sar



Precision based on aspect assessments of all searches by topic - Rutgers University
Searchers: a=s001 b=s002 c=s003 d=s004 e=s005 f=s006 g=s007 h=s008 i=s009 j=s010 k=s011 l=s012 m=s013 n=s014

Figure 7: Revised Precision Results.

M., and Zull, C. (1967). A field experimental approach to the study of relevance assessments in relation to document searching. In (Final Report to the National Science Foundation). Cleveland, OH: Case Western Reserve University. (NSF Contract No. C-423)

Saracevic, T. (1975). Relevance: a review of and a framework for the thinking on the notion in information science. *Journal of the American Society for Information Science*, 26, 321-343.

Shaw, W. M. J., Wood, J. B., Wood, R. E., and Tibbo, H. R. (1991). The cystic fibrosis database: Content and research opportunities. *Library and Information Science Research*, 13, 347-366.

6 Appendix A - Specification of the TREC-5 interactive task

The TREC interactive task will use a single set of topics selected for this purpose. The topics are given below. They are taken from the new topics designed for the main TREC adhoc experiment; however, the task given to the searchers is not the same as that required for the main experiment, and a few of the topics have been slightly modified to suit the interactive task set. Searches will be made on the same set that is being used for the main adhoc experiment.

Each searcher will search some topics on the local system under investigation, and some on the NIST ZPRISE system. ZPRISE is a basic query-formulation-and-ranked-retrieval system, and should be set up in a standard configuration, specifically without the relevance feedback facility available in the latest version, but with a facility for recording user selection of items. It may be set up locally or used remotely at one of the other sites involved in the experiment.

6.1 Experimental design

Topics come in four blocks (B1-B4) of three topics each, that is 12 topics altogether. A minimum of

four searchers is required (S1-S4), each searching one block of three topics on each of two systems: ZPRISE as control (C) and the local experimental system (E). Each searcher therefore does six topics in total. The design is:

	B1	B2	B3	B4
	--	--	--	--
S1	E	C		
S2		E	C	
S3			E	C
S4	C			E

6.2 Topic category and task

The topics which have been chosen require the discovery of a range of different aspects for satisfactory resolution, and the task given to searchers is defined in these terms.

The task is then to conduct an interactive search with the following specification: "Your job is to identify as many aspects of each of the following topics as you can, in twenty minutes. You should find and save at least one document for as many different aspects as possible. 'Aspects' can mean, for instance, different developments in a field, or different methods, or different countries, or different opinions. To identify precisely the type of aspect, you should read carefully the 'Description' of the topic."

Participants may need to refine the specification given, and/or provide additional guidance, in their instructions to their searchers. Any such refinement/guidance should form part of the report.

6.3 Data to be recorded

It will be necessary for the system and/or the searcher to record and report on the progress and outcome of the search, in various ways. There follows a series of notes on specific items that need to be recorded; below is a partial specification of the reporting format.

Documents selected: the searcher's selection (choice) of items for the final output list must be identified. This selection and the time taken form the basis for the summary evaluation (see below).

Time taken: the elapsed (clock) time taken for the search, from the time the searcher first sees the topic until s/he declares the search to be finished, should be recorded. It is assumed that the interactive search takes place in one uninterrupted session. If a session is unavoidably interrupted, it is recommended that it be abandoned and the topic given to another searcher.

Sequence of events: all significant events in the course of the interaction should be recorded. The events listed below are those that seem to be fairly generally applicable to different systems and interactive environments; however, the list may need extending or modifying for specific systems.

- Timing of events: it may be necessary to record the times of individual events in the interaction (see below).
- Intermediate search formulations: if appropriate to the system, these should be recorded.
- Documents viewed: "viewing" is taken to mean the searcher seeing a title or some other brief information about a document; these events should be recorded.
- Documents seen: "seeing" is taken to mean the searcher seeing the text of a document, or a substantial section of text; these events should be recorded.
- Terms entered by the searcher: if appropriate to the system, these should be recorded.
- Terms seen (offered by the system): if appropriate to the system, these should be recorded.
- Selection/rejection: documents or terms selected by the user for any further stage of the search (in addition to the final selection of documents).

6.4 Reporting formats

The evaluation (see below) is separated into two stages: a summary evaluation, involving a small number of measures defined here, and an analytical evaluation, which may involve a number of different measures not all of which have yet been defined. The

"sparse" reporting format defined here is intended to provide the minimum data required for the summary evaluation; the "rich" format is to provide the additional information needed for the analytical evaluation.

"Sparse" format: a list of the identifiers of the selected documents for each topic, together with the elapsed (clock) time of the search.

"Rich" format: for each topic, the sequence of events as indicated above, and perhaps the times of events. The specification of this rich format will be based on that discussed for TREC-4; it is likely to require further interaction among the groups taking part, to ensure that all groups can comply.

Two further items should be reported. A full narrative description should be given of the interactive session for one specified topic (274). As indicated above, any further guidance and/or refinement of the task specification given to the searchers should also be reported.

6.5 Summary evaluation

The summary evaluation measures will be aspect-recall (that is, proportion of the aspects of the topic covered), precision (according to the usual relevance judgements), both measured for the selected set, and elapsed search time, as defined above. These constitute a triplet of measures, to be taken together. Some account will need to be taken of the results on the control set of searches on ZPRISE.

6.6 Analytical evaluation

Participants are invited to consider ways of measuring performance, and/or of diagnosing the effects of system features or other variables.

6.7 Topic blocks

B1: 256i, 274i, 293i
B2: 260i, 264i, 286i
B3: 254i, 284i, 292i
B4: 255i, 258i, 299i

The order of topics within a block is not significant.

7 Appendix B

7.1 General instructions

These are the general instructions to be given to each searcher at the beginning of the interaction with both the control and the test system.

Imagine that you have just returned from a visit to your doctor during which it was discovered that you are suffering from high blood pressure. The doctor suggests that you take a new experimental drug, but you wonder what alternative treatments are currently available. You decide to investigate the literature on your own to learn what different alternatives are available to you for high blood pressure treatment. You are only interested in finding one document for each of the different treatments for high blood pressure. You find and save a single document that lists four treatment drugs. Then you find and save separate documents that discuss the use of calcium, regular exercise, biofeedback, and the snakeroot plant as possible alternative treatments. In all, you have identified eight different aspects for this topic in five documents.

Now we would like you to identify as many aspects as possible for each topic that will be presented to you. You will be given 20 minutes to search for each topic's aspects. Please save one document for each of the aspects that you identify. If you save one document that contains many aspects, try not to save additional documents that contain only those aspects, unless a document contains additional aspects as well.

Carefully read each description and narrative for each topic because the interpretation of "aspects" changes from topic to topic. For example, aspects can refer to different developments in a field, to different instances in which an event can occur, or to different kinds of treatments – as it did in our example above.

Do you have any questions about

- what we mean by aspects
- the way in which you are save non-redundant documents for each aspect?

7.2 Specific instructions

These are the specific instructions for each individual topic. The searcher should be given the text of each topic exactly as follows, each topic text at the beginning of each search. Timing of the search begins with the searcher's receipt of the topic.

7.3 Topic text

7.3.1 Block B1

Number: 256i

Topic: Negative Reactions to Reduced Requirements for College Undergraduate Core Studies

Description: Colleges for a long time have been reducing their requirements in such core subjects as history, literature, philosophy, and science. Criticism of this trend has occurred.

Please save at least one document that identifies EACH DIFFERENT criticism of this trend. If one document discusses several criticisms, then you need not save other documents that repeat those aspects, since your goal is to identify the different criticisms that have been made.

Narrative To be relevant, a document will provide negative opinions/facts concerning the fact that colleges have reduced their basic requirements for the granting of degrees to undergraduates.

Number: 274i

Topic: Electric Automobiles

Description: What are the latest developments in the production of electric automobiles?

Please save at least one document that identifies EACH DIFFERENT recent development in the field

of electric automobiles. If one document discusses several developments, then you need not save other documents that repeat those developments, since your goal is to identify the different ones that have been discussed.

Narrative: The economic feasibility of electric automobiles appears to be limited by a number of factors, including the limited range of operation between recharges of batteries. What progress has been made in addressing these factors?

Number: 293i

Topic: Evacuation of U.S. Citizens by U.S. Military

Description: Evacuation of U.S. Citizens by U.S. Military since the year 1900.

Please save at least one document that identifies EACH DIFFERENT instance of evacuation in this century. If one document discusses several historical instances, then you need not save other documents that mention those evacuations, since your goal is to identify the different evacuations that have occurred.

Narrative: The U.S. Military has many times been called upon to evacuate non-combatant U.S. personnel from Foreign Countries. A relevant document will identify instances, since 1900, where U.S. military personnel have been called on to evacuate non-military U.S. citizens from possible chaotic and dangerous situations around the world. Evacuation of military personnel should not be considered relevant unless accomplished in concert with the evacuation of civilians.

7.3.2 Block B2

Number: 260i

Topic: Evidence of human life

Description: What evidence is there of the existence of human life in the New World 10,000 and more years ago?

Please save at least one document that identifies EACH DIFFERENT source or type of evidence of human life in the New World 10,000 and more years ago. If one document discusses several cases of such evidence, then you need not save other documents that mention those same types or sources, since your goal is to identify the different types and sources of evidence that have been found and discussed.

Narrative: To be relevant a document would show evidence that humans existed in the New World 10,000 and more years ago. The evidence could range from human hairs, tools, fossils, etc. to bones of animals indicating human presence. Relevant documents would have to specify the evidence and the age suspected, as well as the basis for any theory presented. Information on the Clovis Culture would be relevant. Archaeology digs would probably be related in some way to each relevant document.

Number: 264i

Topic: U.S. Citizens in Foreign Jails

Description: Identify instances where U.S. citizens have been or are being held in foreign jails since the year 1900.

Please save at least one document that identifies EACH DIFFERENT case of a U.S. citizen being held in a foreign jail. If one document discusses several such cases, then you need not save other documents that mention those cases, since your goal is to identify all of the different cases of U.S. citizens being held in foreign jails.

Narrative: With International travel becoming more commonplace, many travelers, whether due to ignorance of the law or with criminal intent, are ending up in foreign jails. Relevant documents will discuss the country involved, the reason for the seizure and jailing, and the sentence, if convicted. U.S. intervention, if any, would also be of interest.

Number: 286i

Topic: Paper Cost

Description: Identify the factors which have led to the cost of paper rising.

Please save at least one document that identifies EACH DIFFERENT factor that has led to increase in the cost of paper. If one document discusses several such factors, then you need not save other documents that mention those same explanations, since your goal is to identify all of the different factors which have been discussed or suggested.

Narrative: In the last year or so the publishing industry is said to have faced a 40% increase in the cost of paper. What factors have led to this price rise? For example, Is there a shortage of materials? Has the cost of processing risen? Have higher taxes been imposed?

7.3.3 Block B3

Number: 254i

Topic: Non-invasive procedures for persons with heart ailments

Description: What kinds of medications or procedures other than heart surgery have been used to treat heart ailments?

Please save at least one document that identifies EACH DIFFERENT medication or procedure which has been used. If one document discusses several such medications or procedures, then you need not save other documents which discuss these individual medications or procedures, since your goal is to identify the different ones that have been used.

Narrative: A relevant document will report/discuss those cases in which persons diagnosed with heart ailments were treated with medications and/or techniques such as angioplasty, stents, lasers, arthroctomy (roto router) etc., in place of surgery. Also advantages of non-invasive procedures over

surgery and comparative studies which show any disparity in longevity when either procedure is used.

Number: 284i

Topic: International Drug Enforcement Cooperation

Description: Identify instances where it is shown that international cooperation is taking place in an effort to combat the worldwide drug problem.

Please save at least one document that identifies EACH DIFFERENT instance. If one document discusses several cases of international cooperation, then you need not save other documents which mention those same instances, since your goal is to identify the different cases of international cooperation.

Narrative: Drugs are a critical problem being experienced throughout the world. International Cooperation will be required if this problem is to be alleviated. Examples of relevant documents would be those related to shared information regarding shipments across international borders as well as sharing of information with various prosecutors and other law enforcement personnel.

Number: 292i

Topic: Worldwide Welfare

Description: Identify social programs for poor people in countries other than the U.S.

Please save at least one document that identifies EACH DIFFERENT social program in EACH DIFFERENT COUNTRY. If one document discusses several programs in different countries, then you need not save other documents which mention those same programs, since your goal is to identify the different such programs which exist.

Narrative: To be relevant a document would identify a welfare program in a foreign country and explain how it works to aid citizens who have little or no income. It would include those who can't work

because of a disability and people who have the extra burden of small children. The document should indicate how these people are supported or not supported. A relevant document should identify the source of the monies used to support such welfare programs.

7.3.4 Block B4

Number: 255i

Topic: Environmental Protection

Description: Name countries that do not practice or ignore environmental protective measures.

Please save at least one document that identifies EACH DIFFERENT COUNTRY without environmental controls. If one document discusses several such countries, then you need not save other documents that also mention those nations, since your goal is to identify all of the different countries which do not have environmental controls.

Narrative: Nations that do not practice or ignore environmental protective controls degrade the progress other nations have made in this vital area. There are international efforts to protect the environment. The actions of some countries are of some concern, however, because they may be ignoring efforts to conserve and protect the world's resources. The objective of this topic is to identify countries that do not have environmental controls.

Number: 258i

Topic: Computer Security

Description: Identify instances of illegal entry into sensitive computer networks by non-authorized personnel.

Please save at least one document that identifies EACH DIFFERENT case of illegal entry into a computer network. If one document discusses several

cases or instances, then you need not save other documents which mention the same ones, since your goal is to identify the different instances of illegal entry.

Narrative: Illegal entry into sensitive computer networks is a serious and potentially menacing problem. Both 'hackers' and foreign agents have been known to acquire unauthorized entry into various networks. Items relevant to this subject would include but not be limited to instances of illegally entering networks containing information of a sensitive nature to specific countries, such as defense or technology information, international banking, etc. Items of a personal nature (e.g. credit card fraud, changing of college test scores) should not be considered relevant.

Number: 299i

Topic: Impact on local economies of military downsizing

Description: What kinds of impact on local economies have been caused by U.S. military downsizing and base closure at U.S. and foreign soil locations?

Please save at least one document that identifies EACH DIFFERENT KIND of impact on local economies. If one document discusses several impacts, then you need not save other documents which discuss the same consequences, since your goal is to identify the different types of impacts due to military base closure or military downsizing.

Narrative: Relevant documents would contain information on the specific type of impact on the local economy of one or more U.S. military base closures.

8 Appendix C - Reporting of results to NIST

1. Sparse format - TWO files from each site

(a) Search file

One line for EACH SEARCH, each line containing the following blank-delimited items from left to right:

1. Unique site ID

2. Search ID

Participant's choice but ids for searches using PRISE must begin with the string "PRISE."

3. Searcher ID - participant's choice

4. Block number (e.g., "B1")

5. Block sequence number for this searcher ("1" or "2")

6. TREC topic number

7. Elapsed time - number of secs., fractions truncated

Clock time from the moment the searcher sees the topic until the moment the searcher indicates the search is complete.

(b) Documents file

One line for each document in a given search result, each line containing the following blank-delimited items from left to right:

1. Chronological sequence number (e.g., "1", "2") Use number of last time saved if saved multiple times.

2. Search ID (from search file)

3. TREC document identifier (DOCNO)

NOTE: Reported data items listed within each line must NOT contain whitespace.

2. Rich format

a) ASCII file - no wider than can be printed in landscape mode using 10pt. font on 8.5x11 inch paper.

3. Full narrative description for topic 274: Electric Automobiles

a) ASCII file - no wider than can be printed in portrait mode using 10pt. font on 8.5x11 inch paper.

4. Description of any further guidance/refinement of the task specification given to the searchers

a) ASCII file - no wider than can be printed in portrait mode using 10pt. font on 8.5x11 inch paper.

9 Appendix D - Instructions for NIST aspect assessors

You will be presented with a series of topics. Each is a description of an information need - roughly, a "question" to which there may be multiple "answers". For example, if the topic is:

Topic 1

Description: There has been a great deal of effort expended in recent years in the search for the ways to treat high blood pressure. These have included investigations of a variety of means, not all of which involve new drugs.

Answers: each different treatment for high blood pressure available today.

Narrative: Relevant documents will discuss the actual use of various means, traditional or non-traditional, for the treatment of hypertension in humans today.

then answers might be the following:

- use of calcium
- regular exercise
- biofeedback
- the Mediterranean diet
- etc.

You will also be presented with a set of documents, each of which should be relevant to the topic because it discusses one or more of the different possible answers to the question posed by the topic.

You are to do the following separately for each topic:

1. read the topic carefully.
2. read each of the documents for the topic and gradually:

3. create a list of the possible answers found somewhere in the documents. You should select a short phrase to describe each answer.
4. determine which documents contain which answers.

So, if you were given the example topic on treatments for hypertension and a set of 8 documents (call them: D1, D2,... D8,) you might end up with the following:

Topic: 1 - Answers

1. use of calcium
2. regular exercise
3. biofeedback
4. the Mediterranean diet

Answers contained by each document

- D1: 1,2
- D2: 1,3
- D3: 1
- D4: 3
- D5: 1,4
- D6: 2
- D7: 4
- D8: none

In this example we are acting as though Document D1 discusses the use of calcium and regular exercise to treat hypertension but none of the other "answers". Document D4 mentions the use of biofeedback as a treatment. Document D8 contains NO discussion of any answer to the question posed by the topic - this is possible but unlikely.

Feel free to read the documents as many times as needed and to revise your list of answers as well as the list of which answers are provided by which documents.

Do you have any questions about what you are supposed to do ?

10 Appendix E - Summary Evaluation

10.1 City University, London

Summary Results for Each Search

Search id	NOTE: See the associated system/site										
Searcher id	report for information about what										
Block number	portion of the interactive task										
Block sequence number	was executed to produce these data.										
Topic											
Elapsed time (secs)											
Num docs saved											
Precision(%)											
Aspectual Recall(%)											
Summary vector of aspects covered											
(1=covered; leftmost = #1)											
254_and	a	B3	1	254	1246	7	71	78	111001111		
254_kar	b	B3	1	254	1206	14	79	33	110000001		
255_bri	c	B4	1	255	1308	4	75	23	10111100100000000000000000000000		
255_col	d	B4	1	255	1207	6	0	4	0000000000000000100000000000		
256_cla	e	B1	1	256	1291	6	0	14	0000001		
256_lis	f	B1	1	256	1213	1	0	14	0000001		
258_bri	c	B4	1	258	1302	8	62	33	100011001001101001000000		
258_col	d	B4	1	258	835	7	29	17	100001001000000001000000		
260_ala	g	B2	1	260	1215	3	67	67	101011		
260_sar	h	B2	1	260	1294	2	50	17	000010		
264_ala	g	B2	1	264	1269	7	86	29	100100010000000110		
264_sar	h	B2	1	264	1278	7	100	24	000101001000000100		
274_cla	e	B1	1	274	1249	13	69	91	11111111110		
274_lis	f	B1	1	274	1222	13	85	73	11111110100		
284_and	a	B3	1	284	1303	6	67	24	1001101000000100000100000		
284_kar	b	B3	1	284	1223	9	44	36	1001101010110010001000000		
286_ala	g	B2	1	286	1278	7	86	44	110100001		
286_sar	h	B2	1	286	1249	14	86	78	111100111		
292_and	a	B3	1	292	1253	2	50	6	00110000000000000000000000000000		
292_kar	b	B3	1	292	1375	6	33	16	00000001000100001100100000000000		
293_cla	e	B1	1	293	1216	7	14	17	100000		
293_lis	f	B1	1	293	1236	4	0	0	000000		
299_bri	c	B4	1	299	1040	7	29	47	001100011100101		
299_col	d	B4	1	299	1129	3	0	0	0000000000000000		

10.2 Rutgers University

Summary Results for Each Search

Search id	NOTE: See the associated system/site report for information about what portion of the interactive task was executed to produce these data.									
Searcher id										
Block number										
Block sequence number										
Topic										
Elapsed time (secs)										
Num docs saved										
Precision(%)										
Aspectual Recall(%)										
Summary vector of aspects covered										
(1=covered; leftmost = #1)										
254i-002-2	b	B3	2	254	1103	7	100	33	111000000	
254i-003-1	c	B3	1	254	1095	12	75	78	111110101	
254i-005-2	e	B3	2	254	1260	13	15	44	101001100	
254i-007-1	g	B3	1	254	1075	3	67	33	101000001	
254i-011-1	k	B3	1	254	1108	9	78	44	101010001	
254i-012-2	l	B3	2	254	1246	7	71	44	111000001	
254i-014-2	n	B3	2	254	1043	7	71	44	101010100	
255i-003-2	c	B4	2	255	1141	16	38	81	1101111101111111111111000	
255i-004-1	d	B4	1	255	1280	1	100	15	001111100000000000000000000	
255i-006-2	f	B4	2	255	1214	5	20	19	10000000000000000100000111	
255i-008-1	h	B4	1	255	1242	2	100	23	111111000000000000000000000	
255i-009-2	i	B4	2	255	1330	2	100	19	001111100000010000000000000	
255i-012-1	l	B4	1	255	1289	6	50	15	110100000000010000000000000	
256i-001-1	a	B1	1	256	1315	5	20	29	1000001	
256i-004-2	d	B1	2	256	1222	5	0	14	0000100	
256i-005-1	e	B1	1	256	1132	3	0	0	0000000	
256i-007-2	g	B1	2	256	1013	10	40	71	1111010	
256i-009-1	i	B1	1	256	1163	2	50	29	1001000	
256i-010-2	j	B1	2	256	617	11	0	14	1000000	
256i-013-1	m	B1	1	256	1092	2	50	43	1011000	
258i-003-2	c	B4	2	258	1182	28	64	79	1111111111111111101011000	
258i-004-1	d	B4	1	258	1138	7	29	38	111011110000100000100000	
258i-006-2	f	B4	2	258	1225	7	43	38	111011110000100100000000	
258i-008-1	h	B4	1	258	1094	17	53	67	111101101001100111001111	
258i-009-2	i	B4	2	258	1231	5	80	54	110111100000110010001111	
258i-012-1	l	B4	1	258	1214	10	90	67	111111110001110010001111	
260i-001-2	a	B2	2	260	1149	7	57	100	111111	
260i-002-1	b	B2	1	260	1061	2	100	67	101011	
260i-006-1	f	B2	1	260	1333	1	100	50	001011	

260i-008-2	h	B2	2	260	1211	8	62	83	111011
260i-010-1	j	B2	1	260	1195	12	17	50	001011
260i-011-2	k	B2	2	260	1178	2	50	50	001011
260i-013-2	m	B2	2	260	751	3	33	33	100010
260i-014-1	n	B2	1	260	1304	0	0	0	
264i-001-2	a	B2	2	264	1147	9	78	47	10111101100100000
264i-002-1	b	B2	1	264	1131	6	83	35	10010101100010000
264i-006-1	f	B2	1	264	1217	3	67	6	10000000000000000
264i-008-2	h	B2	2	264	1182	11	100	29	10010101000000010
264i-010-1	j	B2	1	264	1344	15	87	47	00110101101010100
264i-011-2	k	B2	2	264	1127	13	100	47	01010101101010100
264i-013-2	m	B2	2	264	830	3	100	18	00010100100000000
264i-014-1	n	B2	1	264	1298	15	100	71	01110111110011101
274i-001-1	a	B1	1	274	1095	10	100	73	11111110100
274i-004-2	d	B1	2	274	1047	8	100	64	11111100100
274i-005-1	e	B1	1	274	1385	6	83	73	01111110110
274i-007-2	g	B1	2	274	712	14	100	73	11111110100
274i-009-1	i	B1	1	274	1247	5	100	73	11111110100
274i-010-2	j	B1	2	274	1243	36	94	100	11111111111
274i-013-1	m	B1	1	274	1191	6	100	64	11111100010
284i-002-2	b	B3	2	284	1157	15	60	48	1100011000010110001111001
284i-003-1	c	B3	1	284	1188	22	50	52	1101101001010110001110100
284i-005-2	e	B3	2	284	1157	13	69	44	1101111000000100011101000
284i-007-1	g	B3	1	284	1015	9	44	24	1000100000000110010010000
284i-011-1	k	B3	1	284	1018	17	41	44	1101101000011101011000000
284i-012-2	l	B3	2	284	1201	18	67	56	0111101110010111101000010
284i-014-2	n	B3	2	284	818	10	80	36	0001101000010111011000000
286i-001-2	a	B2	2	286	1152	28	64	89	110111111
286i-002-1	b	B2	1	286	939	6	100	44	110100001
286i-006-1	f	B2	1	286	1226	4	50	44	011110000
286i-008-2	h	B2	2	286	1121	14	7	44	110110000
286i-010-1	j	B2	1	286	1013	9	44	56	111100010
286i-011-2	k	B2	2	286	1251	11	82	56	110110001
286i-013-2	m	B2	2	286	590	5	60	44	110101000
286i-014-1	n	B2	1	286	1087	15	73	56	110110001
292i-002-2	b	B3	2	292	1110	4	50	19	00000001000110000000010000110000
292i-003-1	c	B3	1	292	1067	19	21	56	101111101101110000000101111001100
292i-005-2	e	B3	2	292	1209	22	36	41	00010001001100110011010000111001
292i-007-1	g	B3	1	292	1254	16	31	47	00010001001100111111110000111000
292i-011-1	k	B3	1	292	1213	19	47	47	00010001001100110111110000111010
292i-012-2	l	B3	2	292	1212	22	32	44	10010001001100110011110000111000
292i-014-2	n	B3	2	292	945	11	45	38	00010001001100110011010000111000
293i-001-1	a	B1	1	293	1206	5	0	0	000000
293i-004-2	d	B1	2	293	1357	0	0	0	
293i-005-1	e	B1	1	293	1261	11	45	100	111111
293i-007-2	g	B1	2	293	1337	1	100	17	000100

293i-009-1	i	B1	1	293	1136	5	20	17	000100
293i-010-2	j	B1	2	293	964	6	0	0	000000
293i-013-1	m	B1	1	293	1088	2	50	17	100000
299i-003-2	c	B4	2	299	1195	18	39	73	111111111100010
299i-004-1	d	B4	1	299	1251	3	33	47	101111101000000
299i-006-2	f	B4	2	299	1337	6	0	7	000000000000100
299i-008-1	h	B4	1	299	1140	9	44	87	101111111111011
299i-009-2	i	B4	2	299	1236	2	50	40	101111100000000
299i-012-1	l	B4	1	299	1157	7	57	60	111111110000010

11 Appendix F

11.1 Comments of the relevance assessor on topic 284

International Drug Enforcement Cooperation

Scan terms: World-wide problem international cooperation law enforcement personnel prosecutors borders shipments

Comments: The drug culture has had a profound impact upon persons, families, the country, education and the world economy.

International cooperation regarding a coherent policy to confront, interdict and neutralize drug traffickers has been hampered by a combination of political rivalries, jealousies, official corruption and inefficiency. In contrast, organized international criminals have become sophisticated and adept at extending its drug operations across countries and continents and of manipulating the world economic system.

The United States is the major country targeted by the drug traffickers because of the demand and personal financial resources. Other countries, however, particularly in Europe, are now feeling the effects of the drugs within their cultures. If the supply and availability of drugs is to be curtailed or limited, it will require the cooperation of all countries – growers, manufacturers and users alike. Only through a concerted international effort can drugs be identified, interdicted and destroyed.

Those items considered to be relevant were those in which there was cooperation, collaboration and collusion between the U.S. and another country(ies) to locate, interdict and destroy international shipments of drugs and incarcerate those persons involved in the trafficking.

Items included: cooperation among countries to combat international drug trafficking; UN and drug companies exchanging information on illegal drug market-

ing; agreement among Thailand, Burma, Malaysia, Pakistan and China to work with the United States in cutting opium production; Hong Kong asking for and receiving assistance from the DEA (in the form of helicopters), to identify, locate and dismantle labs making cocaine; U.S. giving speed boats to Santo Domingo to interdict the flow of drugs; Bolivia and Brazil agreeing to join forces to fight the drug cartel; and five countries and two continents cooperating to crack a drug smuggling ring in Milan, Italy.

Not considered relevant were those items which reported on internal actions to interdict, subvert, destroy or apprehend drug dealings within a country, regardless of the origin of the drugs or the nationality of the traffickers.

11.2 Comments of the aspect assessor on topic 284

Criteria employed in identifying an article included statements regarding: - international meetings at which plans were discussed to stop drug trafficking - information regarding drug shipments - cooperation between two or more countries to interdict drug shipments - training of police/agents in foreign countries to target drug maintenance - providing (by U.S.) of vehicles, helicopters, materials, herbicides to destroy crops, warehouses storing the drugs

Specific actions by an individual country to interdict a drug shipment, apprehend drug smugglers within the confines of a country: i.e. unilateral action, was not considered relevant

11.3 Author's review of document judgements for topic 284

11.3.1 AP880213-0180 Relevance error?

"While the two leaders (Reagan and de la Madrid) met, senior officials from both countries held talks of their own, focussing on foreign policy, trade, drugs and law enforcement."

11.3.2 AP880222-0072 Relevance error?

The document mentions meeting between Reagan and de la Madrid and cites statement by de la Madrid about the need to attach production, distribution, and consumption - no specific mention of cooperation but the two leaders were meeting and considering the issue so it seems relevant.

11.3.3 AP880401-0280 Relevance error?

"We're talking the same language," said white House official Donald Ian MacDonald of increasing U.S. - Soviet cooperation on drugs. ... regional organizations in several parts of the world have been formed to tackle the problem.

Two clear references to international cooperation.

11.3.4 AP880406-0185 Relevance error?

The document describes cooperation between Honduras, the Dominican Republic, and the United States in the arrest of a drug dealer.

11.3.5 AP880409-0127 Relevance error?

The document U.S. Attorney General's tour of Latin America to assess anti-drug trafficking efforts and in particular meetings with Peruvian officials involved in the anti-drug fight.

11.3.6 AP880411-0168 Relevance error?

The document quotes Edwin Meese as saying: "Peru has been one of the leaders in the use of herbicides for eradicating coca plants. We have been cooperating with Peru in these experimental studies."

11.3.7 AP880415-0060 Relevance error?

The document quotes Mexico's federal attorney general as saying: "Mexico is undertaking a plain effort of international cooperation in the campaign against drug trafficking." There are also references to financial assistance from the United States to Mexico for anti-drug efforts.

11.3.8 AP880415-0165 Relevance error?

"It [the Mexican government] also cooperates with the U.S. Drug Enforcement Administration, which maintains an office and agents here [Mexico]."

11.3.9 AP880519-0173 Relevance error?

The document reports on the arrest and conviction of Carlos Lehder Rivas, a major Columbian drug dealer. Attorney General Meese quoted calling the case "an excellent example of international cooperation."

11.3.10 AP880519-0201 Relevance error?

Essentially the same information as AP880519-0173

11.3.11 AP880520-0105 Relevance error?

Essentially the same information as AP880519-0173

11.3.12 AP880617-0255 Relevance error?

The document reports statements about goals and expectations of discussions at an economic summit about drug trafficking, drug-money laundering, "a general discussion of the whole drug problem..."

11.3.13 AP880811-0234 Relevance error?

"Soviet customs agents helped with the best tip of the year. They told the Mounties that a drug ring planned to ship 11,277 pounds of hashish from Kabul, Afghanistan, through Leningrad to Montreal. The hash was seized on arrival and three Canadians are now behind bars."

11.3.14 AP880927-0103 Relevance error?

"Treaties with six countries calling for cooperation in narcotics prosecutions and other criminal matters were approved by the Senate Foreign Relations Committee on Tuesday...."

11.3.15 AP881011-0041 Relevance error?

The document reports on the Inter-regional Narcotics Eradication Air Wing. "[U.S.] civilian-piloted gunships will join Peruvian narcotics police in raids to destroy cocaine laboratories, warehouses and airstrips."

"The State Department began its international anti-drug campaigns in 1978 when Congress offered Mexico four helicopters to spray marijuana and opium poppies with the herbicide paraquat. The State Department now maintains about 120 aircraft involved in drug efforts in Mexico, Columbia and Burma."

11.3.16 AP881112-0073 Relevance error?

The document mentions the existence of U.S. State Department anti-drug programs in Bolivia and Columbia without spelling out the details. It also mentions a \$10 million [U.S.] narcotics program program in Columbia.

11.3.17 FT911-2671 Relevance error?

The document describes three international agencies to be merged into one U.N. organization and says these agencies "implement and monitor treaties on legal drug production and run intervention programmes to try to change the economic and cultural bases of Third World countries producing opium and coca leaves."

11.3.18 FT921-6988 Relevance error?

The document talks about a U.S. - Latin America drug summit and mentions U.S. aid and military presence in drug-producing states.

11.3.19 FT922-9654 Relevance error?

The document describes a "debt-for-drugs" scheme being investigated by the U.N. pursuant to a motion passed by the U.N. Commission on Narcotics Drugs. According to the document, most of the nations which would be asked to forgive debt in exchange for drug-fighting efforts Third World countries say they can't otherwise afford, are opposed to the idea. So this may not be any cooperation here.

11.3.20 FT924-10092 Relevance error?

The document mentions a U.N. pilot project in Laos which required a change in the U.N.'s approach to include "planned, co-ordinated management of U.N. agencies, funding bodies and international aid from developed countries..."

11.3.21 FT934-10491 Relevance error?

"The new [U.S.] efforts, say officials, would build on improved co-operation already evident, particularly with Bolivia and Columbia in the Andes."

11.3.22 FT942-10977 Relevance error?

The document refers to "Umopar, a para-military jungle unit trained by the DEA, the U.S. Drug Enforcement Agency, to eliminate the Bolivian drug trade."

11.3.23 WSJ910529-0062 Relevance error?

"The Bush administration pursues an integrated policy that attacks illegal drugs and money laundering on the international front. Its anti-money-laundering strategy seeks to establish a network of countries joined together These programs will facilitate international cooperation...."

11.3.24 WSJ920113-0038 Relevance error?

The document mentions Hong Kong's participation in the Financial Action Task Force, "a 26 nation group.... The FATF last year issued recommendations aimed at destabilizing trafficking organizations ..."

11.3.25 WSJ920204-0078 Relevance error?

The documents cites several instances in which Panama under General Noriega assisted the United States in the war on drugs.

11.3.26 WSJ920214-0116 Relevance error?

"President Fujimori's commitment. U.S. support for the decriminalization of coca farmers, reports of President Bush's personal backing, and similar support from some European governments and the United Nations triggered ... the drafting of initial plans for private investment" in Peru to allow coca farmers to switch to other crops.

11.4 Comments of the relevance assessor on topic 292

Many documents mentioned the fact that a country has a social program for the poor and unemployed but gave no insight as to how the program worked. These were not considered relevant. Some non-relevant documents supplied information on how poor a country was but did not indicate that they had any welfare system. Other documents identified the source of income for their welfare program but did not include sufficient information on how this money supported the recipients. Generally when I found information on a country including how much funding was spent for a specific type of recipient, I made it relevant. Some documents included enough information on its unemployment problem to make it relevant. If a document stated that a welfare recipient lost their benefits because they attained a specific amount of savings or income, I made it relevant. A document was relevant when it stated how a country supported its welfare program, i.e. taxes. While FT data appeared to be the best source for relevant documents, the CR and FR data had none.

11.5 Comments of the aspect assessor on topic 292

1. If the document treats government programs (pensions, health, housing, et al) as a unit, I listed them as ONE answer.

2. Of the first 11 documents 7 were on the U.S.; therefore did not provide answers since the topic defined as "in countries other than the U.S." (I checked to make certain there was not incidental or comparative reference to other countries.

3. Ref 1 above: Since several docs deal with reductions in funding of government social programs, they may mention specific programs but by lack of description they tend to lump the programs all together. In these cases I have listed a single "answer", e.g. "South Africa - government programs"

4. Doc 18 (FT931-10916) is essentially a think piece with only fleeting references to the British welfare system and how it does (or doesn't) work.

5. Doc 34 (FT934-12800) Demographic data on countries was not considered an "answer" unless it was related directly to a welfare benefit in a particular country.

6. Doc 62 (FT944-6725) The article does not really say that the Social Action Program has accomplished anything, but refers to it as "launched", "embraced", and "extended". I counted it as an answer.

Spanish and Chinese Document Retrieval in TREC-5

Alan Smeaton

School of Computer Applications
Dublin City University

Ross Wilkinson

Department of Computer Science
Royal Melbourne Institute of Technology

April 5, 1997

1 Multilingual Document Retrieval in TREC

The TREC-5 conference was the third year in which document retrieval in a language other than English was benchmarked. In TREC-3, 4 groups participated in an ad hoc retrieval task on a collection of 208 Mbytes of Mexican newspaper text in the Spanish language. In TREC-4 there were 10 groups who participated, once again in an ad hoc document retrieval task on the same Mexican newspaper texts but with new topics. In TREC-5 there was a change of document corpus and new topics for the Spanish ad hoc retrieval task and a corpus of documents and topics to support ad hoc retrieval in the Chinese language was introduced for the first time. There were 7 groups who submitted runs for the Spanish track and 10 who submitted results for Chinese.

The corpus of texts used in the TREC-5 Spanish language task was approximately the same size as the one used in TREC-3 and TREC-4 but differed in that there was a more consistent use of accented characters and it was European Spanish as opposed to Mexican Spanish. This slightly affected the morphological processing of word forms.

In the Chinese language each character represents at least a complete syllable, rather than a letter as in other languages. Many characters are also single syllable words. The total number of characters is therefore quite large and somewhat ill defined. A literate adult would typically recognise at least 5-6,000 characters. The various modern standards define between 10-12,000 characters, although if early and ancient literature is included the number rises to approximately 100,000. Chinese is agglutinating – there is no space between consecutive characters, except perhaps, at the end of a sentence. Thus to perform retrieval in Chinese, the basis has to be characters unless the text is pre-segmented into words.

2 Retrieval Task

The retrieval task for the Spanish and Chinese tracks are exactly the same as the standard ad-hoc task in TREC. A given database of texts and a fixed set of topics are supplied. The task is to return a ranked list of 1,000 documents for each of the topics. For each topic, at least one run using only the description part of the topic is encouraged. The topics were supplied in both English and either Spanish and Chinese. Either the English could be used so that cross lingual retrieval could be explored, or the language of the document collection could be used for monolingual experiments.

For Spanish, a 173,950-document collection from the Agence France Presse 1994 newswire was used. It was 308 Megabytes as raw text. 25 topics were constructed for the Spanish collection and there were on average 100 relevant docs per topic. From the 7 groups whose results were pooled, each topic had an average of 679 documents manually assessed for relevance.

For Chinese, a 164,811 document collection included documents from both the People's Daily and the Xinhua News Agency. There was no segmentation information supplied. It was 170 Megabytes as raw text. 28 topics were constructed for Chinese. There were on average 93 relevant docs per topic for Chinese.

3 Spanish Results

The 7 groups who took part in TREC-5 Spanish used a variety of approaches for indexing and retrieval. We summarise these approaches before discussing their comparative retrieval effectiveness.

Cornell University

The SMART system used in the mainline submissions from Cornell University was also used in the Spanish track. A simple stemmer which removed common word endings was used to normalise word occurrences. As per Cornell's mainline submission in TREC-5, documents were initially ranked and then re-ranked based on Rocchio relevance feedback assuming documents initial top-ranked were in fact relevant. In addition to indexing by normalised word occurrences, the the Cornell approach also used any pair of non-stopwords occurring sufficiently frequently in the corpus as a *statistical phrase*. What is especially notable about the Cornell submission, apart from retrieval effectiveness, is that the porting effort needed to enable SMART to handle a corpus in Spanish was very small, and this included the development of the word normalisation module.

University of Massachusetts

As with Cornell University, the submission from the University of Massachusetts was much the same as their previous work, in this case a follow-on from a TREC-4 Spanish track approach and using a variation of their mainline TREC-5 work. This was based on combining global analysis and local feedback to generate query expansions based on local context analysis, essentially expanding queries by terms from top-ranked documents if they occur near query term occurrences. This group used a sophisticated word normalisation process developed in their TREC-4 Spanish work.

Rank Xerox Research Center

The Xerox group used a Spanish part-of-speech tagger to identify and index by noun pairs as an alternative to the simple adjacent non-stopword word pairs or “statistical phrases” used by others such as Cornell. The Xerox group also used a linguistically motivated lemmatizer to reduce word occurrences to their root forms, as an alternative to the comparatively crude stemming process used by most IR systems.

The Xerox group were also the first to use the English version of the queries as supplied by NIST, in some of their runs. They used an inflectional morphology stemmer to stem the text of the English version of the topics and automatically translated each resulting word root form into its Spanish equivalent for retrieval using a dictionary lookup. This represents the first attempt in TREC to do cross-language retrieval.

University of California, Berkeley

The Berkeley group had used a morphological stemmer in their TREC-4 Spanish track and this was enhanced for their TREC-5 submission with the addition of a larger list of irregular and regular verb forms. In addition, acronyms were identified and excluded from the stemming process and an attempt was made to correct the inconsistent use of accented characters in documents and in topics by comparing each word occurrence against a massive, unstemmed wordlist. The retrieval algorithm used by the Berkeley group was the same as in their TREC-4 Spanish.

Dublin City University

The group from Dublin City University took part in the Spanish track in order to evaluate the performance of a new stemming algorithm for Spanish developed by Martin Porter. Porter’s Spanish stemmer is essentially built along the same principles as his 1980 stemmer for English; a word ending

is taken off if the remaining stem is of a suitable “length” as determined by vowel-consonant patterns. There is also a list of word prefixes which are not used in calculating a stem’s “length”, e.g. “CON-”. Porter’s stemmer also knows about 500 of the most commonly occurring irregular verbs in Spanish, much more than in English. The retrieval algorithm used by the DCU group was plain vanilla $tf*IDF$ term weighting and document ranking, with no phrase recognition or document partitioning, as used in the mainline ad hoc submission from this group.

George Mason University

Like the group from Cornell, GMU used a two-pass retrieval algorithm incorporating estimated relevance feedback into the second pass. The original topics were used as part of a $tf*IDF$ term weighting to generate a first document ranking and then terms in the top-10 ranked documents were ranked by their $n*IDF$ values and the top 10 of those were added to the query for the second pass retrieval. There was no stemming in the GMU approach though a 500-word stopwords list was used. The implementation of this was on GMU’s relational database prototype.

New Mexico State University

The final group from NMSU concentrated on cross-language retrieval technologies by using the English version of the topic description and automatically translating into Spanish for running against the Spanish documents. The automatic translation processes included an attempt at text disambiguation among multiple senses in a bilingual dictionary, incorporating all translations from the dictionary, and a baseline run of Spanish queries against Spanish documents. The retrieval engine used in all cases was, like most groups, based on $tf*IDF$ term weighting, and also used a Porter-like word stemmer once the dictionary-based translation had been completed.

Conclusions on Spanish Retrieval

Retrieval on Spanish texts threw up some interesting observations for the groups who took part. Even though the corpus was of European as opposed to Mexican Spanish, and the use of accented characters was more consistent, it was still variable across documents. Some groups took a simple approach to this and ignored all accents while for others there was a serious attempt to use and even to correct the accented character use. In Spanish the accents on characters generally does not affect the meaning of the words so we don’t know what effects the special treatment vs. the ignoring approaches have.

For most groups participating in the Spanish track, the track was an add-on, an afterthought, with

a low priority after their mainline TREC submissions and the effort required to alter established systems such as SMART and INQUERY to handle Spanish texts was minimal. Some groups like NMSU and Xerox, however, made it a priority. The amount of effort put in by respective sites is not reflected in the relative performances in terms of precision and recall as approaches which worked well for the English mainline task carried over reasonably well into the Spanish retrieval task also. For groups which had developed sophisticated and effective retrieval techniques for the mainline task their reward for taking part in the Spanish task was the knowledge that their techniques worked well in Spanish also.

Despite the small number of participating groups, it is difficult to make cross-system performance comparisons, even when retrieval uses the same documents and topic descriptors. An expected result is that the longer forms of the queries incorporating the narrative as well as the description, yields better averaged performance but that does not tell us anything new. Many of the groups developed new approaches to word normalisation and it would have been interesting to compare directly, the normalisation approaches taken by Cornell, Xerox, Berkeley and Dublin, for example. This would have been the interesting and the language-specific issue in the Spanish track. Unfortunately, and this is a symptom of TREC in general, each group which develops a new technique for any part of the retrieval process must then wrap that technique within the IR system used at their site and other IR parameters such as stopword lists, term weighting, document length normalisation, document scoring, and so on, are all different from site to site and this prevents direct cross-site comparisons. Thus the only scientifically credible comparisons we can make are among the runs submitted from a single site, where the other retrieval parameters are not variable. The alternative to this would be an unworkable version of TREC.

The two groups which tried cross-language retrieval, Xerox and NMSU, both found that their cross-linguistic results were not as good as using the original Spanish form of the queries. The NMSU group claim to have recaptured 70% using their best English-Spanish, whereas the Xerox performance drop is more severe. Another observation from the Xerox work is that adding linguistically derived noun phrases/noun pairs improved retrieval effectiveness over not adding, but only slightly, so this raises the question of whether it is it worth the effort. This is a big question in IR research anyway.

4 Chinese Results

The 10 groups who took part in TREC-5 Chinese generally explored the use of words vs. n-grams and methods of manually modifying queries. Some work was also done on retrieval methods particularly appropriate to Chinese retrieval. We summarise these approaches before discussing their comparative retrieval effectiveness.

City University

The experiments at City University used the Okapi system for their Chinese retrieval experiments. They tried both a character based retrieval and a word retrieval. Words were discovered using a greedy algorithm using a 70,000 word dictionary. With both character and word approaches, the use of phrases were explored. A number of probabilistic relationships were investigated based on the relative probability of a phrase appearing given that both constituents have appeared.

Claritech Corporation

The Claritech used the Clarit system for both n-gram character based and word based approaches. Words were discovered using a dynamic programming algorithm that segmented into minimal numbers of words using a 100,000 word dictionary and heuristics. Their automatic experiments showed that word based and bi-gram approaches were roughly comparable and that both were better than single character retrieval. Claritech also explored manual retrieval by introducing new terms and introducing Boolean constraints on documents to be used for feedback. Both of these techniques provided improvements. It was interesting to note that feedback that usually works well in most other circumstances failed to provide gains for manual retrieval.

Cornell University

Cornell approached Chinese retrieval with no Chinese expertise but a very good retrieval system – the SMART system. They achieved very good results. They approached the task by using character based retrieval augmented with character bi-grams. Expansion techniques worked well.

George Mason University

Only single characters were used in the George Mason approach. Their base run again showed that reasonable performance can be obtained using characters only. They then applied term expansion by selecting the top 10 terms from the top 10 documents. This gave good improvement. However having a person ensure that only relevant documents were used in selecting terms for expansion gave no improvement.

Information Technology Institute

The Information Technology Institute ran both automatic and manual experiments. The automatic experiment was a weighted word based ranked approach. Queries were then modified manually using fuzzy Boolean constructs. This approach gave very substantial improvements.

Queens College, CUNY

The approach taken here was to apply a combination of dictionary and statistical techniques to detect 2, 3 and occasionally 4 character words. The aim was to obtain good indexing features rather than “correct” segmentation. The also tried using both characters and words which gave a small improvement.

Swiss Federal Institute of Technology

ETH used fully automatic techniques using both single characters and all possible bi-grams. Term expansion was then applied. There were no official runs submitted.

Royal Melbourne Institute of Technology

The RMIT approach was to use several automatic runs based upon characters and words found by dictionary methods. No run was successful.

University of California, Berkeley

The Berkeley group put a lot of effort into building a good dictionary of 140,000 words to use to automatically segment the text. They then applied their standard ranking methods. Queries were then modified with several hours of manual effort with considerable improvement.

University of Massachusetts

A hidden markov model was used to segment text in the University of Massachusetts approach. The resulting queries used characters, groups of characters, and words. Experiments were then conducted using the Local Context Analysis approach to term expansion. Performance was better when expansion terms had lower weights.

General Remarks

This was the first year of this track, so it was not possible for groups to optimize their techniques on past data. Nevertheless it does appear to be the case that the best n-gram approaches, including single character approaches were comparable to the best word based approaches. Most techniques for improving retrieval performance using term expansion worked well – as they had in the comparable English experiments. It was noticeable how much improvement occurred when groups manually modified the queries. The best runs were obtained as a result of manual modification.

Report on the TREC-5 Confusion Track

Paul B. Kantor SCILS, Rutgers Ellen Voorhees NIST

Abstract

For TREC-5, retrieval from corrupted data was studied through retrieval of single target documents from a corpus which was corrupted by producing page images, corrupting the bit maps, and applying OCR techniques to the results. In general, methods which attempted a probabilistic estimation of the original clean text fare better than methods which simply accept corrupted versions of the query text.

1 History

The confusion track originated at an informal meeting held during TREC-3, stimulated by interest in the potential of various schemes for retrieval based on imperfect OCR applied to scanned legacy texts. The guiding idea was that even an imperfect translation of the image into text might support effective retrieval, especially if the retrieval were based on text representations not dependent upon the identification of terms. Participants in this first meeting included Mark Damashek (NSA), David Grossman (GMU), Fritz Nordby (then at Paracel), and Paul Kantor, who was selected, on the “grey hair” principle, to serve as spokesman.

At that first meeting a range of methods for approaching the problem were considered, including use of overlapping n-grams of varying length, and efforts to reverse engineer the transition probabilities of the corruption operator. It was agreed that the process should involve a corruption algorithm whose transition matrix was known only to the TREC organizers, and not to the participants. The tenor of this discussion led to the informal name “corruption track”, which was subsequently Bowdlerized to the more presentable “Confusion Track”.

For TREC-4 the track was managed in a very low-key fashion (by PBK), and lost one of its originators when Nordby moved to another position. The participants, whose results will not be summarized here, obtained results which were not judged to be exciting. The most successful results were reported by Buckley at Cornell [1].

2 The TREC-5 Task

For TREC-5 the confusion track used a particular type of retrieval problem called *known-item searching*. A known-item search is a retrieval task that simulates a user seeking a particular, partially-remembered document in the collection. In contrast to a more standard retrieval search where the goal is to retrieve/rank the entire set of documents that pertain to a particular subject of interest, the goal in the known-item search is to retrieve one particular document.

Known-item searching is well-suited to the task of retrieving corrupted data. When document content is corrupted, low-frequency words such as proper nouns and technical terms are the most affected. Yet low-frequency words are high-content-bearing words, and are precisely the words likely to be used to locate a specific document. Thus known-item searches exercise the parts of

- Use of solar power by the Florida energy office.
- Excessive mark up of zero coupon treasury bonds.
- I am looking for a document about the dismissal of a lawsuit involving Adventist Health Systems.
- I am looking for theft data on the Chevrolet Corsica.
- efforts to establish cooperative breeding programs for the yellow crowned amazon parrot.
- morphological similarities between different populations of saltwater crocodiles.

Figure 1: Example known-item topics from the TREC-5 confusion track

the retrieval methodologies that the track is most interested in. As a bonus, the searches do not require relevance assessments. Clearly, this eliminates the need for relevance assessor's time — a critical resource at NIST. It also means the track can run with fewer participants: since TREC uses pooled results to approximate exhaustive relevance assessments, the exhaustivity of the relevance assessments depends on the diversity of the pool and hence on the number of participants.

Participants in the track were asked to rank the top 1000 documents per topic on each of three different versions of the *1994 Federal Register*: the correct copy, a scanned copy that had approximately a 5% character error rate, and a scanned copy that had approximately a 20% character error rate. The 20% error rate version was created by performing OCR on an image that had been downsampled from the original image.

Figure 1 shows some examples of the known-item topics used in the track. The topics were created by five NIST staff members who developed ten topics each. Different authors used different techniques to construct their topics (using an index of the collection to find unique words, starting with “interesting” documents and adding conditions to the topic to ensure uniqueness, etc.), but the authors did *not* specifically pick words that they thought would be difficult for the OCR process. The authors made every effort to ensure that only one document was a legitimate answer to the topic, and there have been no reported problems with this assessment.

3 Overview of Retrieval Schemes

Five broadly different retrieval methods were applied in TREC-5. They form a progression in terms of the detail with which they attempt to discern the correct text underlying the corrupted version. The reader is urged to consult the respective papers by the individual participants found elsewhere in this proceedings for more details about the approaches.

Rutgers SCILS APLab: The Rutgers APLab group used Unix utilities to search for any string, in the corrupted text, matching any n-gram defined by stopping at word boundaries and using a sliding window of width 5 within longer words. After removal of stop words (using the SMART list) match was defined using a wild card character in regular expressions, which matches any set of 0,1, or 2 characters.

Each term in the query was expanded into all the “expanded” patterns that meet these rules. Thus “cat” is searched as all of “cat, .at, c.t, ca..”. The corrupted text was processed line by line. Every match contributed one point to the score of the current line. The final score for a text was the average of the line scores. Note that this scheme gives added weight to uncorrupted texts. Thus “cat” in the text would match all four of the patterns.

Australian National University (ANU): In this approach queries were “corrupted”, based on corruption errors discovered in a small sample of clean and 5% degraded texts. In effect, this expanded the query by the addition of likely corrupt terms. The span scoring method was then applied to corrupted texts and expanded queries to produce the final results.

George Mason University (GMU): The GMU approach used one-step retrieval, based on overlapping 4-grams, including term boundaries. Queries were expanded by the addition of the most frequent n-grams occurring in the top 10 documents, and the top 20 n-grams were added to the query with weight 0.4. Document-query similarity scores were computed using a cosine measure, with an inverse document frequency metric.

CLARITECH Corporation (CLARIT): In the CLARITECH approach, stochastic methods were applied to the documents to correct corrupted words on a sentence-by-sentence basis. Federal register data from 1988 and 1989 was used to estimate word frequencies, and word-word transition (bi-gram) probabilities. Correction was applied *only* to words that did not match the lexicon (call these *c-words*). For each such c-word, up to 200 candidates were listed and ranked in order of “probability to match the corrupt word”. The top 10 candidates to each c-word were retained for sentence processing. This processing seeks to minimize the total stress, over the sentence, of the transitions between consecutive words. Thus CLARIT produced a single sentence-based-maximum likelihood assignment for every c-word, as a basis for further retrieval. Further retrieval was accomplished using the standard CLARIT indexing and retrieval.

Swiss Federal Institute of Technology (ETH): The ETH group made use of several devices. Document score was computed using a pivot method (which controls the effect of overly long documents), and feature contributions which include $f(\phi, d)$, the observed frequency of a feature ϕ in the document d (queries are also treated as documents); $d(\phi)$, which measures the prevalence of feature ϕ in the collection; and f^* , a corrected estimated frequency. The latter is determined in three steps. First, the document is divided into overlapping slots, which might contain the feature. A slot is used for further computation if it contains at least a fraction P of the feature’s characters. The number of slots such that the edit distance between the slot contents and the feature is less than 20% of the length of the feature is determined. The probability that a feature appears in a slot is set to a nominal value for each matching slot, and then summed over all the slots in the document. The sum is then multiplied by a constant which makes the estimate of feature frequency more accurate, as determined by regression analysis applied to a set of 100 documents used in clear, 5% and 20% corrupted forms. The features used are (Porter) stems, including the preceding white space.

In practice, this detailed calculation was carried out only for a set of 2000 documents for each query, retrieved by straight n-gram screening, using $n=4$ for 5% and 3 for 20% corrupted data.

In sum, the ETH method gives a document credit for all of the features which have a “sufficiently close match” to the noisy text as it is presented. All features are words or initial substrings of words, and the method could have trouble with corruption of the word separation characters.

As these brief descriptions show, the methods vary in their treatment of the query, and of the corrupted texts. They appear to form a progression in the following sense:

Rutgers expanded the terms appearing in the query by a 5-gram sliding window with each character replaced with any set of 0, 1 or 2 characters. The 5-grams did not cross word boundaries. Retrieval ranking was based on the average number of hits per line of text. This probably discriminates much too strongly against long documents. Performance was poor.

ANU expanded queries based on corruption errors found likely in a study of a sample of corrupted text. Thus additional terms (which might in principle be words in a lexicon) were added to the query.

GMU resolved both query and documents into overlapping 4-grams, judged to be more resistant to corruption, and required an exact match. Special stop-lists of 4-grams were constructed. Queries were expanded by a method based on preliminary retrieval from the corpus, resulting in the addition of new 4-grams.

These three methods represent expansion of the query, in an effort to include or match corrupted forms that either might (Rutgers), or could (ANU), or sometimes do (GMU) happen under the corruption observed. Results for the first of these methods were relatively weak; the second method was not applied to the most severely corrupted data, and the third exhibited somewhat surprising performance detailed below.

The remaining two methods sought to “expand” or “clarify” the corrupted texts.

CLARIT used statistical methods to replace each non-word by a word which makes the entire resulting sentence most likely in some well-defined sense. Each non-word is replaced by exactly one word.

ETH, in effect, replaced each “slot” (which might be occupied by a word in the corrupted text) by a vector of candidate words, each of which is permitted to contribute to the computed similarity to the question. This is, in principle, a wider expansion of the corrupted text, since the second ranked candidate can enter the computation in this method, but not in the CLARIT method.

4 Retrieval Results

Participants were asked to submit a ranking of the top 1000 documents for each topic. The runs are evaluated based on the rank given to the target document; no “partial credit” was given for retrieving documents similar to the target.

4.1 Evaluation Measures

Several different evaluation measures are given for each system in the confusion results in Appendix A. The first measure is the Raw Ranks table. This table gives the rank at which the known item was retrieved for each of the three versions of documents for all 49 topics¹. A document that was not retrieved at all in the top 1000 documents was assigned a rank of 2000.

The “mean-rank-when-found” and the “mean-reciprocal-rank” are given in the final rows of the Raw Ranks table. The mean-rank-when-found is the mean rank at which the known item was found averaged across all topics that retrieved the known item in the top 1000 documents. This measure gives an easily-interpreted idea of how well the retrieval methodology ranks the known item if it finds it at all. (When the average is computed over all topics, this measure is also known as *expected run length*.)

¹Topic 29 had to be dropped from the evaluation. Some input files were mistakenly truncated when producing the degraded versions of the text, so all three collections were restricted to the smallest of the three sets. One of the omitted documents was the target item for topic 29.

The mean-reciprocal-rank is the mean of the reciprocal of the rank at which the known item was found, averaged over all the topics, and using 0 (not $1/2000$) as the reciprocal for topics that did not retrieve the known document. Unlike the mean-rank-when-found measure, this measure penalizes runs that did not retrieve a known item in the top 1000 while minimizing the difference between, say, retrieving a known item at rank 750 and retrieving it at rank 900. It is also bounded between 1 and 0, inclusive, so the measure is interpretable without knowing how many documents were ranked. Indeed, since there is only one relevant document per query, the reciprocal rank of that document is the precision at that document, and therefore it is the average precision of the query as well (average precision is the precision averaged over all relevant documents of the query). Average precision is a frequently used measure in the other parts of TREC, so “mean-reciprocal-rank” is comparable with other retrieval methods.

A histogram of the ranks at which the known item was found is given in the second table in the appendix. It gives the number of topics for which the item was not retrieved at all, was found in the first ten ranks, was found in rank 11–100, and was retrieved but ranked greater than 100.

A graphical representation of the results is given below the histogram. The graph plots the cumulative percentage of queries whose known item was found by each rank. For example, if the value of the curve is 27 at rank 15, then 27% of the topics retrieved their known item at rank 15 or lower.

4.2 Comparative Performance

As mentioned above, expected search length is not a good measure to use to compare systems because the largest contributions come from the nominal positions assigned to those documents which were not received. On the other hand, its reciprocal, which increases for better systems, does not have this problem. We will use a Generalized Retrieval Operating Characteristic [2] to obtain a more detailed view of comparative performance.

If we consider the cumulated value delivered by a set of ranked lists to be proportional (with constant v) to the number (G) of target documents found (out of a total of S target documents), and cost as proportional (with constant c) to the number of documents which must be examined before reaching them, the corresponding measure of value has three terms. The first term is the value of all documents found; the second term is the cost of finding those documents; and the third term is the cost of not finding the remaining documents.

$$\begin{aligned} V &= vG - c \sum_{i \text{ found}} r(i) - 1000(S - G)c \\ &= G(v + 1000c) - c \sum_{i \text{ found}} r(i) - 1000S \end{aligned}$$

Hence systems would be ranked according to:

$$G(v + 1000c) - c \sum_{i \text{ found}} r(i)$$

This is the same as ranking them according to:

$$G\left(\frac{v}{c} + 1000\right) - \sum_{i \text{ found}} r(i)$$

In other words, the relative importance of finding a document at all, compared to the importance of placing found documents high in the list, depends in an unavoidable way on the cost assigned to examining documents. Thus there is no single measure which covers all reasonable opinions about this parameter.

On the other hand, one may extend the idea of a Generalized Retrieval Operating Characteristic to compare systems. The extension is to imagine that all retrieved lists are perused in parallel,

and to ask how many of the documents have been found, when r documents have been examined on each list. This performance curve may be calculated directly from the reported lists as follows. Let $r(j)$ be the rank of the sought item, for topic j . We can compute the cumulated number of documents examined up to each “hit”. In effect, we imagine that for a set of S topics, there are S analysts who work in parallel. Each examines the next document in the list for her problem. As soon as she finds the desired document, she stops working on this task. If she reaches the end of the list she stops working. The process continues until all the analysts have stopped working. We plot the number of good documents that have been discovered against the number of documents that have been examined in all rounds prior to its discovery.

If the curve for one scheme lies everywhere above the curve for another then, at least for this set of target documents, it delivers greater value, whatever the value (v) assigned to good and cost (c) assigned to bad documents. If neither curve is always above the other, then we cannot definitely state that one scheme is to be preferred to another. As with other measures currently used in information retrieval evaluation, the statistical significance (confidence levels, confidence intervals, etc.) this type of comparison is not known, particularly for the case of multiple comparisons.

The overall confusion track results roughly parallel the order of generality of the nets cast by the methods as described in Section 3. Figure 2 uses the rank of the target document as the abscissa. In Figure 3 we show the results using the economically more meaningful measure w =[cumulated total number of items examined]. We show only the performance achieved by each team on the 20% degraded materials. Conveniently, almost every teams’ better effort dominated its weaker effort, in the sense described above. (The exception is ETH, for which the poorer scheme does eventually surpass the better scheme, measured against the work involved.) In Figure 4 we show only the results for each team’s better effort. ANU is not represented because this group did not submit a report for the 20% degraded material. Rutgers submitted only one report, as shown here.

Study of Figure 4 reveals that, first of all, the simple pattern matching scheme (*rutcf1*) does most poorly. About 20 target items are found in the first 1200 or so examined, and after that progress is minimal. Initially *CLCON20* and *gmu96v21*, with radically different philosophies, perform about equally. But at somewhere around the 2000th item examined, the n-gram based method continues its slow climb, while the single term assignment method begins to level off.

The *ETHD20P* method, which permits multiple interpretations of a slot in the document, climbs early to a striking advantage, and nearly dominates the other methods. However, there is a small regime, corresponding to the recovery of some 4 or 5 documents near the end of the run, where the *gmu96v21* method briefly pulls above the multiple interpretation method. The difference in this region is probably not statistically significant, but it does eliminate the possibility of a clean dominance ordering being reported from this particular trial.

Note that in preparing these figures, we have followed the TREC philosophy that one good document is as good as another, and have not considered which specific target documents were turning up at each particular point on the curve. There is no barrier in principle to doing this analysis, which might prove interesting. In particular, if some schemes do well on some targets, and others do well on others, and *it is possible to tell them apart prior to retrieval*, specific assignment of schemes could produce better performance. Even if it is not possible to tell them apart, some variant of data fusion might produce results superior to those achieved by single schemes.

5 Summary

Most teams reported one or more counter-intuitive results at the conference, some of which may have been clarified by the time of the final paper which appears in this bound volume.

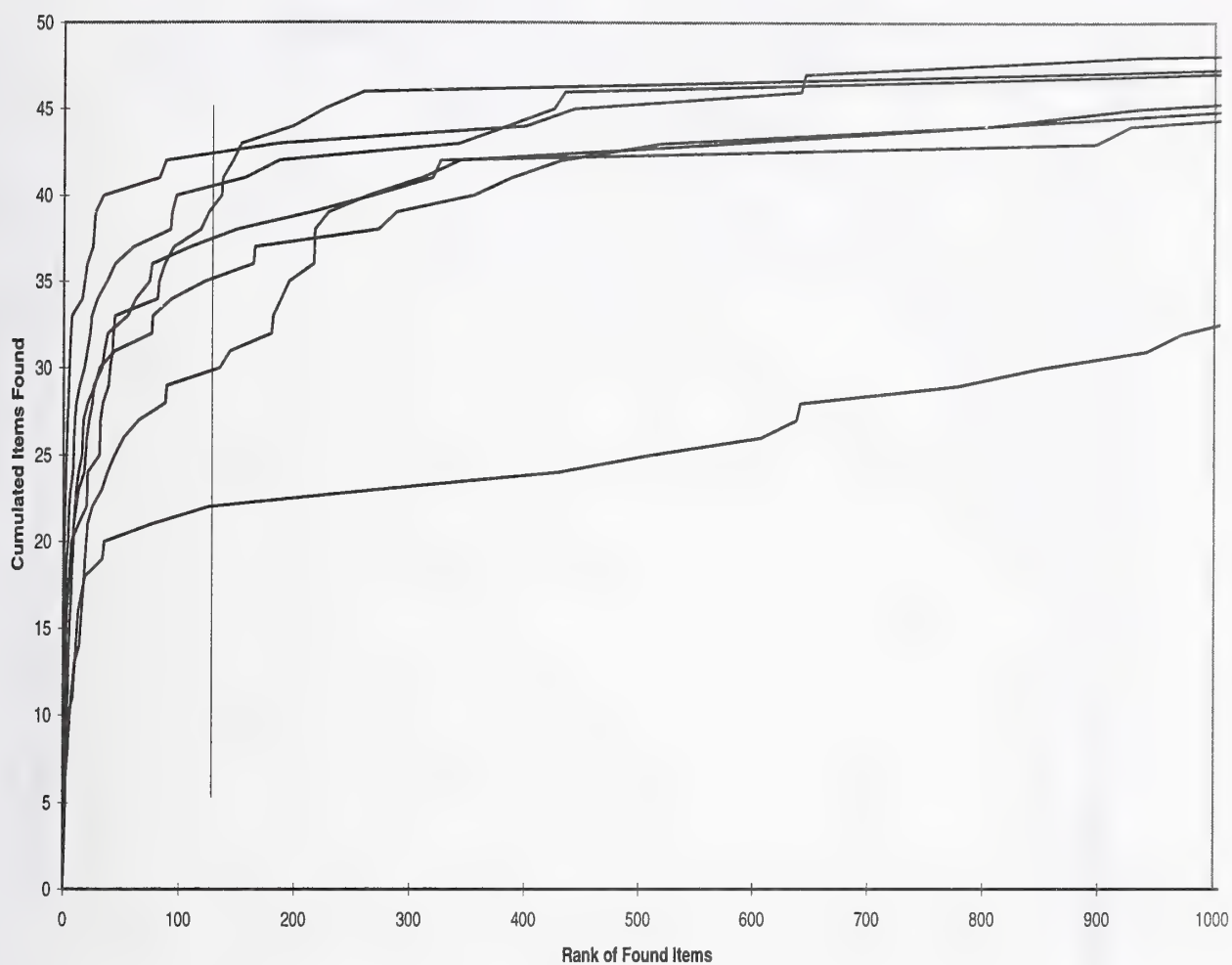


Figure 2: Cumulated relevant documents as a function of the rank in the lists of documents examined. Data are for 20% corrupted text. The systems can be traced by their intersections with the vertical line. In descending order they are: ETHD20P; ETHD20N; gmu96v21; CLCON20; CLCON20F; gmu96v22; rutcf1

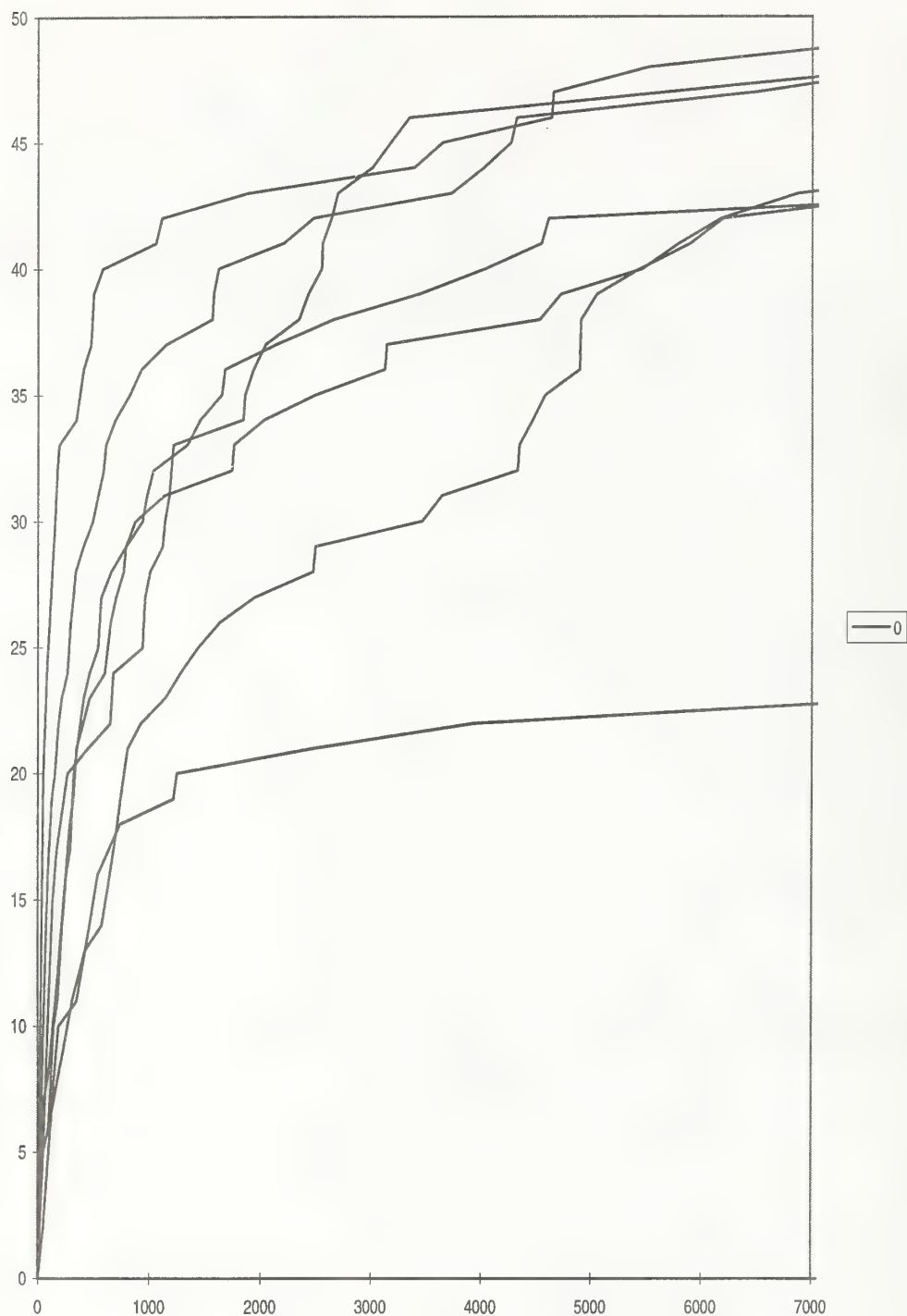


Figure 3: Cumulated relevant documents as a function of the total number of documents examined. The same systems as in Figure 2 are shown here. The number of documents examined is a non-linear function of the rank at which items are found. This causes a change in the appearance of the curves. Note that the crossing of *gmu96v21* and *ETHD20P* covers a much shorter range when presented in this way. Data are for 20% corrupted text.

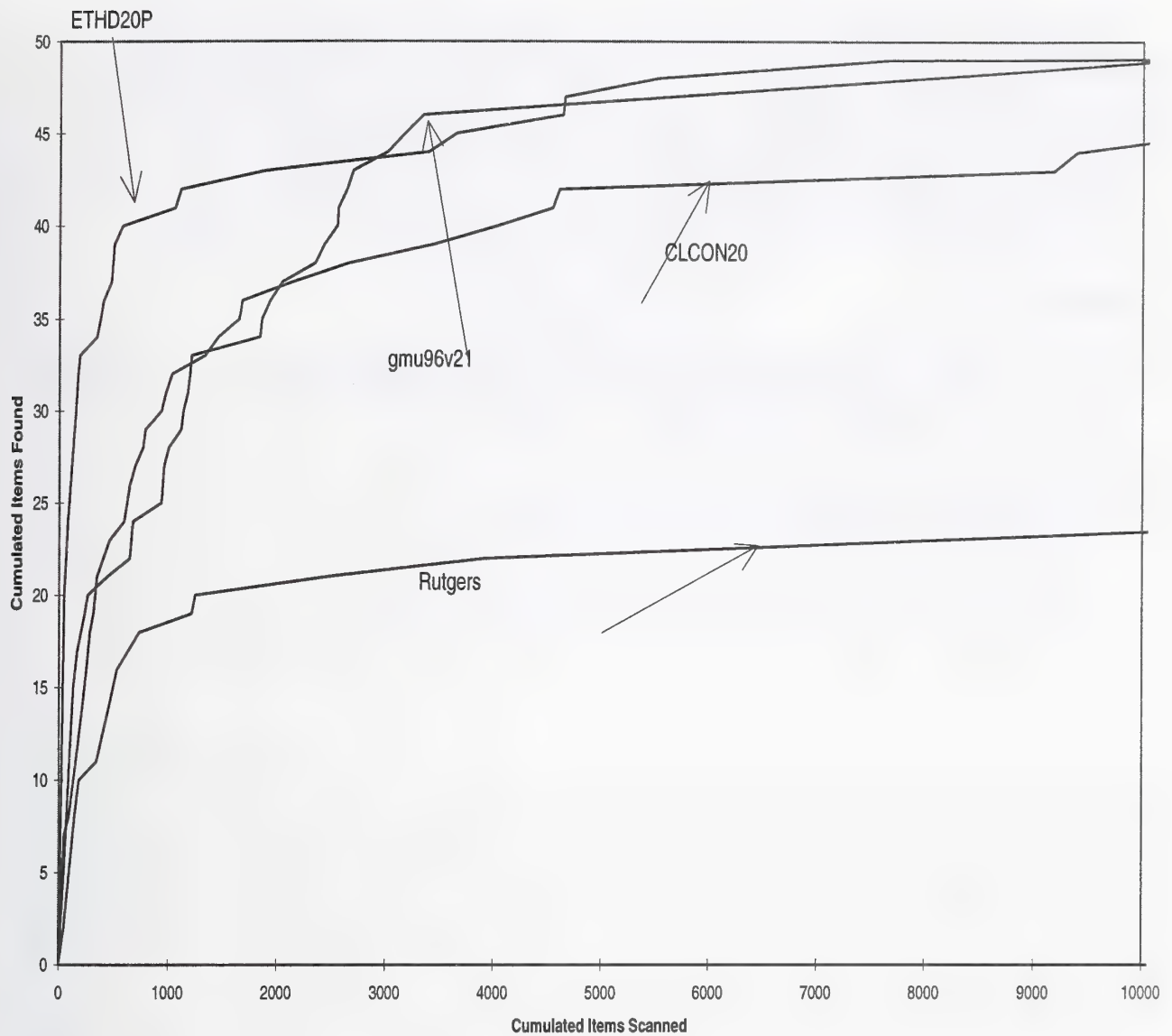


Figure 4: Cumulated relevant documents as a function of the total number of documents examined. Only the dominant best system from each group is shown. Data are for 20% corrupted text.

For example, the ETH initial screening system using 3-grams with Lnu.ltu weighting was substantially more effective than the GMU scheme using 4-grams and cosine weighting. It is not known how much the choice of matching function contributes to this difference, and how much is due to the length of the n-grams. Both the CLARIT team and the GMU team found that their (different) methods of query expansion did not help at all, and made performance worse.

Based on these issues, and the problems noted in the workshop papers, there is still a great deal to be understood about interaction of the diverse approaches used by the participants.

References

- [1] Chris Buckley, Amit Singhal, Mandar Mitra, and (Gerard Salton). New retrieval approaches using SMART: TREC 4. In *Proceedings of the Fourth Text REtrieval Conference (TREC-4)*, pages 25–48, 1996. NIST Special Publication 500-236.
- [2] Paul B. Kantor. Non-linear utility functions in information retrieval. Technical Report APLab Technical Report, SCILS, Rutgers University, 1997.

The TREC-5 Filtering Track

David D. Lewis
AT&T Labs—Research
600 Mountain Avenue, 2A-410
Murray Hill, NJ 07974
lewis@research.att.com
<http://www.research.att.com/~lewis>

Abstract

The TREC-5 filtering track, an evaluation of binary text classification systems, was a repeat of the filtering evaluation run in a trial version for TREC-4, with only the data set and participants changing. Seven sites took part, submitting a total of ten runs. We review the nature of the task, the effectiveness measures and evaluation methods used, and briefly discuss the results. Some deficiencies in the evaluation are examined, with an eye toward improving future filtering evaluations.

1 Introduction

The goal of the TREC-5 filtering track was to aid research groups in evaluating their approaches to binary text classification. As usual in the TREC context, this was done by making available a large data set and doing blind, impartial evaluation of submitted results. The design used in the TREC-5 filtering track was identical to that tested with four sites in TREC-4, thus much of this paper is identical to the description of the TREC-4 filtering track [10].

We begin by defining binary text classification and presenting some applications of it. We then discuss a particular binary text classification task, filtering, used in TREC-5. The effectiveness of filtering submissions was evaluated using utility as a measure. The several roles that this effectiveness measure played in the evaluation are described.

The large size of the TREC-5 test data set meant that relevance judgments were of necessity incomplete and effectiveness could only be estimated. We describe in detail two approaches that were tested for estimating the utility of filtering submissions. Stratified sampling breaks up each filtering submission into related groups of documents and takes a random sample from each group to be judged. Pooled sampling instead uses all the judged documents from the routing and filtering tracks, including the stratified samples.

We end by briefly discussing the results of the TREC-5 filtering track, with an emphasis on what was learned about the evaluation methods.

2 Binary Text Classification

By binary text classification systems, we mean information retrieval (IR) systems that decide for each document processed whether the document should be accepted or rejected [9]. What it means to be accepted varies between systems. Some applications that make use of binary text classification are:

- A company provides an SDI (selective dissemination of information) service which filters newswire feeds. Relevant articles are faxed each morning to clients.
- A text categorization system assigns controlled vocabulary categories to incoming documents as they are stored in a text database.
- An “agent” program monitors low content text streams (e.g. Usenet newsgroups) and alerts a user when a relevant message appears.

Note that there is no notion of a user choosing how far to go down a ranking in these systems. The system makes Yes/No decisions about documents, and the user only sees the results of those decisions. This affects the kind of evaluation appropriate for the system, as discussed in the next section.

3 Evaluation for Binary Text Classification

We have discussed evaluation for binary classification at length elsewhere [8, 9], and so here will concentrate on how it differs from the evaluation of ranked retrieval in the main TREC-5 tasks.

Effectiveness measures for ranked retrieval typically have two components. The first is a *cutoff*: a specification of how to divide the ranking into a top part and a bottom part. The top part is considered to be the set of documents retrieved by the system. The second component of the overall effectiveness measure is a *set-based effectiveness measure* which is applied to the retrieved set.

Some examples of these two-component measures are:

- Precision at 0.10 recall : Here the cutoff is the highest point in the ranking above which at least 10% of the relevant documents in the test set occur. The set-based effectiveness measure is precision, the proportion of documents in the retrieved set which are relevant.
- Recall at 0.001 fallout : Similar to the above, but the cutoff is based on fallout and the set-based effectiveness measure is recall.
- Precision at 20 documents : Here the cutoff is based on a fixed position in the ranking (20 documents down from the top). The set-based effectiveness measure is precision.
- R-precision : This is precision at R documents, where the cutoff R is the number of relevant documents in the collection. The set-based effectiveness measure is precision.

More complex measures, such as the average of precision over multiple recall cutoffs are also used. A wide variety of such measures with different cutoffs and different set-based effectiveness measures have been applied to rankings in the TREC evaluations. This might seem to make it difficult for a site to decide how rank documents, since there is more than one measure to optimize. In truth, all these measures can be optimized simultaneously by the simple and obvious strategy of ranking documents by their probability of relevance. Doing so will result in an optimal score under essentially any reasonable measure of ranking effectiveness, a property which has been formalized as the Probability Ranking Principle [9, 13].

In contrast, binary classification systems make the separation into accepted and rejected documents themselves, rather than leaving this up to the effectiveness measure used in evaluation. The binary classification system must choose what separation to make in order to optimize the effectiveness measure used. Doing so optimally means that the effectiveness measure must be known in advance. Since binary classification systems do not rank the accepted set, the effectiveness measure should be a set-based one. Since the size of the submitted set is under the control of the system, the effectiveness measure used in evaluation must be able to assign an effectiveness to a set of any size, including the empty set.

It is still desirable in the filtering context to test the ability of systems to satisfy varying user preferences (e.g. high recall vs. high precision), but this should not be done by submitting a single ranking and letting the evaluation program pick cutoffs. Instead, a family of effectiveness measures can be used to capture different user preferences. For each measure used, the filtering system produces a separate set of documents appropriate to that measure, and that same measure is used to evaluate the set of documents.

4 Filtering: A Binary Classification Task for TREC

The mainline tasks for TREC-1 through TREC-5, routing and ad hoc retrieval, require participants to submit ranked lists of documents, which are then evaluated using ranking-oriented effectiveness measures. The number of documents submitted is defined in advance, so the ability of systems to pick the number of documents to submit is not tested.

The TREC-5 filtering track addresses this limitation. This section describes the rationale for the evaluation, the evaluation's structure, and the effectiveness measures used.

4.1 Why Filtering?

The main motivation for the filtering track is the increasing number of IR applications requiring binary text classification (see Section 2). The track should help developers of these applications learn about relevant techniques from the research community, and let researchers compare and evaluate their approaches.

A second motivation is that the demands of the filtering task may encourage the development of IR methods with other desirable properties. For instance, accurately estimating the probability of relevance of documents is useful not only in filtering [7, 9, 5], but also for self-monitoring of effectiveness [9], estimating the number of relevant documents [9], and selection of training data

[11].

Finally, we hope that a binary classification task will attract a broader range of researchers and approaches to TREC. The requirement that TREC results be ranked makes it awkward for approaches that are not ranking-oriented to be tried [6, 14]. These approaches include boolean querying by human experts, as well as the use of binary classifiers (e.g. decision trees) produced by machine learning techniques.

4.2 Structure

The structure of the TREC-5 filtering evaluation was as follows. We combined each of 50 TREC-5 topics with each of 3 set-based effectiveness measures, to produce 150 descriptions of user needs. The topic portion of the user need indicates the kind of information sought by the hypothetical user, while the set-based effectiveness measure captures the hypothetical user's tolerance for different kinds of mistakes. The set-based effectiveness measures were based on a notion of utility, as described in Section 4.4.

The topics were the same ones used in the main TREC-5 routing evaluation. However, as with the routing evaluation, data for one topic was accidentally omitted from judging. Therefore the filtering track results are based on 49 topics, with three effectiveness measures for each, yielding 147 descriptions of user needs. In the routing evaluation an additional 4 topics for which no relevant documents were found were also omitted when computing effectiveness measures, but these topics were retained for the filtering track evaluation.

For each user need, the system had to make use of the topic and the effectiveness measure to decide whether to accept or reject each test document. The same test documents as the main routing evaluation were used. The submitted set (i.e. the accepted documents) for the user need was then evaluated using the effectiveness measure for that user need. (Actually only a sample from the submitted set was used in the evaluation—see Section 5.)

4.3 Integration with Routing

The TREC-5 filtering and routing evaluations used the same topics and test documents. (See the discussion by Harman and Voorhees elsewhere in the proceedings.) The training data for the topics (i.e. documents judged with respect to the topics in previous TRECs) were also the same in both evaluations.

The evaluations were also similar in that 100 documents from each site's results for a topic went into the pool for judging. In the case of routing, the judged documents were the top 100 from a single ranked list of 1000 documents submitted for the topic. In the case of filtering, the 100 documents were a stratified sample (Section 5.2.1) from the union of the three unranked sets of documents submitted for that topic. Filtering and routing documents were mixed together and treated identically for judging. The expense of running the filtering track was thereby reduced, since there was considerable overlap between filtering and routing submissions.

4.4 The Utility Measures

The family of effectiveness measures used in the filtering track were based on assigning a numeric value or *utility* to each retrieved document [2, 9]. Retrieved relevant documents received a positive utility, and retrieved nonrelevant documents received a negative utility. The total utility of the submitted set for run R_i was:

$$u_i = u_{ai}A_i + u_{bi}B_i$$

where A_i is the number of relevant documents in the submitted set for run R_i and B_i is the number of nonrelevant documents in the submitted set for run R_i . For each run R_i we assume that u_{ai} is the value the user places on receiving a relevant document, while u_{bi} is the value of receiving a nonrelevant document.

Different values for u_{ai} and u_{bi} define different effectiveness measures in the family. The three utility measures used in the filtering evaluation were:

Run	Parameter Values	Effectiveness Measure
R_1	$u_{a1} = 1, u_{b1} = -3$	$u_1 = A_1 - 3B_1$
R_2	$u_{a2} = 1, u_{b2} = -1$	$u_2 = A_2 - B_2$
R_3	$u_{a3} = 3, u_{b3} = -1$	$u_3 = 3A_3 - B_3$

We might imagine run R_1 corresponding to a user who is willing to pay 1 dollar (or pick your favorite currency) to read each relevant document, and loses 3 dollars worth of time if they have to read a nonrelevant document. Therefore, run R_1 requires the filtering system to act in a conservative or high precision fashion. In constrast, run R_3 encourages systems to emphasize recall somewhat more, while run R_2 is in between.

Unlike recall, but like precision, our utility measures take into account only those documents submitted by the system for a run. (It is possible to define utility measures to take into account rejected documents as well [9].) Unlike both recall and precision, the total utility is not normalized to lie between 0 and 1. Indeed, the minimum and maximum achievable utilities can be determined only if the total number of relevant documents in the test set is known. The goal of systems, however, is simply to achieve the highest utility they can.

4.5 Optimizing The Utility Measures

Different effectiveness measures are more or less easy to optimize by different IR techniques. A common approach to binary text classification is to compute a numeric score for a document to be classified, and assign the document to the class if the numeric score is larger than some specified value. Utility measures which are the sum of utilities for individual documents, like the one used in the TREC-5 filtering evaluation, can be optimized by thresholding if the scores computed are monotonic with probability of class membership [9].

In fact, if the document scores are accurate estimates of probability of relevance, the thresholds to use can be derived directly from the effectiveness measure by decision theoretic principles [3, Ch. 2]. For the TREC-5 filtering measures the optimal thresholds on probability of relevance are:

Run	Parameter Values	Threshold On Probability
R_1	$u_{a1} = 1, u_{b1} = -3$	0.75
R_2	$u_{a2} = 1, u_{b2} = -1$	0.50
R_3	$u_{a3} = 3, u_{b3} = -1$	0.25

Of course, these thresholds are optimal only if the probability estimates produced by a system are in fact accurate.

Note that probability of relevance is the same as *instantaneous precision*. That is, if a system estimated a probability of relevance of p for a large set of documents, and those estimates were accurate, then we would expect that set of documents to have a precision of approximately p . This means that the lowest scoring documents submitted for run R_3 by a highly effective filtering system would have an instantaneous precision of around 0.25. Most of the documents in that system's submitted set would be likely to have instantaneous precisions considerably higher than that. The overall precision for a high quality submitted set should therefore typically be much higher than the threshold probability/instantaneous precision. Thus while run R_3 emphasized recall somewhat more than run R_1 or run R_2 , all three TREC-5 filtering runs required, by the standards of typical IR systems, high precision results.

5 Estimating Total Utility from a Sample

Computing the exact total utility for a submitted run requires knowing the value of A_i and B_i for that run. This would require assessing the relevance of every document submitted for that run. Because submitted sets can be of any size, this might require too much work from the relevance assessors. For that reason, total utility for filtering runs was estimated using *samples* of the submitted documents.

Two different sampling and estimation methods were used in the TREC-5 filtering track, as described below.

5.1 Pooling

The first approach to sampling was the usual TREC pooling strategy [4]. This approach assumes that some known pool of documents contains all the relevant documents in the test set. The pool for the TREC-5 filtering task consisted of all documents judged for the topic in the main routing task, plus all documents judged for the topic for the filtering task, as chosen by the stratified sampling scheme of Section 5.2.1. Under the pooling assumption an estimate \hat{u}_i of the total utility, u_i , is easily computed.

The total utility computed in this fashion is only an estimate, because the pooling assumption may be wrong. There may be submitted documents that are relevant but were not judged for this topic. An advantage of the pooled estimate, however, is that the same sample is used for all sites, enabling that sample to be large. The use of the same sample for all sites also eliminates sampling variation between sites. A disadvantage is that the sample is not a random sample, meaning that it

is difficult to tell how accurate the estimated utilities are (see Section 7.1). There is also the danger that pooled sampling penalizes sites which submit atypical yet relevant documents.

5.2 Random Sampling

To see how random sampling could be used to estimate the total utility it is useful to rewrite the formula for total utility as:

$$\begin{aligned} u_i &= u_{ai} \times A_i + u_{bi} \times B_i \\ &= u_{ai} \times p_i N_i + u_{bi} \times (1 - p_i) N_i \\ &= ((u_{ai} - u_{bi})p_i + u_{bi})N_i \end{aligned} \tag{1}$$

where $N_i = A_i + B_i$ is the total number of documents submitted for the run, and $p_i = A_i/N_i$ is the proportion of documents submitted which are relevant (i.e. the precision of the submission). Rewriting the utility measures used in the filtering evaluation in this way gives:

$$u_1 = (4p_1 - 3)N_1 \tag{2}$$

$$u_2 = (2p_2 - 1)N_2 \tag{3}$$

$$u_3 = (4p_3 - 1)N_3 \tag{4}$$

Therefore, if we can produce an estimate \hat{p}_i of the proportion of relevant documents in a submitted set, we can turn that into an estimate, \hat{u}_i , of the utility of the submitted set:

$$\hat{u}_i = ((u_{ai} - u_{bi})\hat{p}_i + u_{bi})N_i \tag{5}$$

One approach to estimating the proportion would be to take a random sample from the submitted set for a topic/run pair, count the number of relevant documents, a , in that sample, and divide by the number of documents, n , in the sample:

$$\hat{p} = \frac{a}{n} \tag{6}$$

This is called *simple random sampling*. A more complex approach to random sampling often has advantages, as described in the next section.

5.2.1 Stratified Random Sampling

Simple random sampling is not the only way to estimate a proportion. The TREC-5 filtering evaluation estimated utilities using *stratified sampling*. In stratified sampling we use additional knowledge about a population to divide the population into groups or *strata* [1, p. 89]. We then take a simple random sample separately from each stratum, estimate the quantity of interest for each stratum, and combine the stratum estimates to get an overall estimate for the population. If the strata are chosen so that items in a stratum are similar to each other, the accuracy of an stratified estimate can be greater than the accuracy of an estimate based on simple random sampling from the whole population.

R_1	R_2	R_3	Stratum Name (h)	Number of Documents in Stratum (N_h)
0	0	0	000	very many
0	0	1	001	many
0	1	0	010	very few or none
0	1	1	011	some
1	0	0	100	very few or none
1	0	1	101	very few or none
1	1	0	110	very few or none
1	1	1	111	few

Figure 1: The test documents submitted by a site can be separated into eight strata, based on which of the three submitted sets, the R_1 set, the R_2 set, and the R_3 set each document appeared in. We indicate presence in the set by a 1, absence by a 0. The comments indicate the relative sizes of the sets in typical filtering track submissions.

For TREC-5, the set of filtering documents submitted by each site for a topic was stratified according to which of the three runs each document was submitted for. By considering all combinations of presence and absence of a document in the three submitted sets, we get 8 strata, as shown in Figure 1.

In most, but not all, cases the submitted sets will be such that the R_1 set is contained in the R_2 set, and the R_2 set is contained in R_3 set. This means that strata 010, 100, 101, and 110 will usually be empty. In general, however, each of the submitted sets can be the union of four strata:

Set for run R_1 : 100, 101, 110, 111

Set for run R_2 : 010, 011, 110, 111

Set for run R_3 : 001, 011, 101, 111

To estimate the proportion p_i of relevant documents in set R_i by stratified sampling, we separately estimate the proportion p_h for each stratum h in the R_i set. We then add up the estimated stratum proportions, weighting them by the relative size of their stratum in the submitted set [1, p. 91]:

$$\hat{p}_i = \sum_{h \in R_i} \frac{N_h}{N_i} \hat{p}_h \quad (7)$$

Here h ranges over the strata that make up the R_i set, N_h is the size of stratum h , N_i is the size of the R_i set, and \hat{p}_h is an estimate of the proportion of relevant in stratum h . Expanding this out for runs R_1 to R_3 gives:

$$\hat{p}_1 = \frac{N_{100}}{N_1} \times \hat{p}_{100} + \frac{N_{101}}{N_1} \times \hat{p}_{101} + \frac{N_{110}}{N_1} \times \hat{p}_{110} + \frac{N_{111}}{N_1} \times \hat{p}_{111} \quad (8)$$

$$\hat{p}_2 = \frac{N_{010}}{N_2} \times \hat{p}_{010} + \frac{N_{011}}{N_2} \times \hat{p}_{011} + \frac{N_{110}}{N_2} \times \hat{p}_{110} + \frac{N_{111}}{N_2} \times \hat{p}_{111} \quad (9)$$

$$\hat{p}_3 = \frac{N_{001}}{N_3} \times \hat{p}_{001} + \frac{N_{011}}{N_3} \times \hat{p}_{011} + \frac{N_{101}}{N_3} \times \hat{p}_{101} + \frac{N_{111}}{N_3} \times \hat{p}_{111} \quad (10)$$

These estimates will be unbiased (see Section 7) if the estimate \hat{p}_h of the proportion of relevant documents in stratum h is unbiased for each component stratum h . As our estimate \hat{p}_h we used the proportion of relevant documents found in a simple random sample from stratum h :

$$\hat{p}_h = \frac{a_h}{n_h}$$

where n_h is the size of the simple random sample taken from stratum h and a_h is the number of relevant documents found in that sample. This \hat{p}_h is an unbiased estimate of p_h [1, p. 51].

5.3 Sample Sizes in Stratified Sampling for TREC-5

To be consistent with the TREC-5 routing evaluation, at most 100 documents were judged from the three sets of documents submitted by a site for each filtering topic. These 100 documents had to be allocated to as many as seven strata (all except stratum 000), as described above. This was done by choosing equal sized samples from all nonempty strata. If all documents from a stratum were used up by this procedure the leftover documents were allocated equally among the other strata, until a total of 100 was reached, or until all documents from the three submitted sets were selected.

6 Stratified Sampling: An Example

Figure 2 displays data on the submitted sets from a hypothetical filtering site for a single topic. (The test set size differs from that actually used in TREC-5.) The R_2 set is bigger than the R_1 set, and the R_3 set is bigger than both the R_1 and R_2 sets. One anomaly is that 10 documents are in R_2 set but not in R_3 set. This might happen due to a mistake by the site, or because documents were retrieved by boolean queries which were not in a strict generalization relationship.

6.1 Estimating Proportion of Relevant Documents

If all submitted documents were judged, then we could compute the true proportion of relevant documents in each submitted set:

$$p_1 = \frac{30}{30 + 10} = .7500 \quad (11)$$

$$p_2 = \frac{112}{112 + 138} = .4480 \quad (12)$$

$$p_3 = \frac{130}{130 + 1110} = .1048 \quad (13)$$

To compute a stratified estimate of these proportions, we assume that simple random samples were drawn from each stratum and judged for relevance, as shown in Figure 2. This gives estimates of the proportion of relevant documents in each stratum, as shown in the last column of Figure 2.

Stratum	Submitted Docs			Samples from Strata		
	Rels	NonRels	True Prop	Rels	NonRels	Est. Prop
000	50	100000	.0005	0	0	–
001	20	980	.0200	1	29	.0333
010	2	8	.2000	2	8	.2000
011	80	120	.4000	10	20	.3333
100	0	0	–	0	0	–
101	0	0	–	0	0	–
110	0	0	–	0	0	–
111	30	10	.7500	23	7	.7667
R1 Total	30	10	.7500	–	–	.7667
R2 Total	112	138	.4480	–	–	.3973
R3 Total	130	1110	.1048	–	–	.1054
Test Set Total	182	101118	.0018	–	–	–

Figure 2: Hypothetical data on a site’s submitted sets for a single topic. We show both the true and sampled values of the number of relevant and nonrelevant documents in each stratum and run, and the corresponding proportion of relevant.

We then combine the stratum estimates, using Equations 8 to 10, to get estimates of the proportion of relevant in each submitted set:

$$\hat{p}_1 = \frac{40}{40} \times \frac{23}{23+7} = .7667 \quad (14)$$

$$\hat{p}_2 = \frac{10}{250} \times \frac{2}{2+8} + \frac{200}{250} \times \frac{10}{10+20} + \frac{40}{250} \times \frac{23}{23+7} = .3973 \quad (15)$$

$$\hat{p}_3 = \frac{1000}{1240} \times \frac{1}{1+29} + \frac{200}{1240} \times \frac{10}{10+20} + \frac{40}{1240} \times \frac{23}{23+7} = .1054 \quad (16)$$

6.2 Estimating Utility of a Submitted Set

If we knew the true proportion of relevant documents in each submitted set (Equations 11 to 13), we could compute the true utility of each set, using Equations 2 to 4:

$$u_1 = (4 \times .7500 - 3) \times 40 = 0.0 \quad (17)$$

$$u_2 = (2 \times .4480 - 1) \times 250 = -26.0 \quad (18)$$

$$u_3 = (4 \times .1048 - 1) \times 1240 = -719.2 \quad (19)$$

If we instead have the stratified estimates of the proportions, we use them to get estimates of the total utility:

$$\hat{u}_1 = (4 \times .7667 - 3) \times 40 = 2.672 \quad (20)$$

$$\hat{u}_2 = (2 \times .3973 - 1) \times 250 = -51.35 \quad (21)$$

$$\hat{u}_3 = (4 \times .1054 - 1) \times 1240 = -717.2 \quad (22)$$

The estimates for the R_1 and R_3 sets are close to the true values, while the estimate for the R_2 set is less close. We can see why by comparing in Figure 2 the true and estimated proportion of relevant for each stratum in the R_2 set. Due to bad luck with our random sample from 011, the largest stratum in the R_2 set, we underestimated the proportion of relevant in that stratum. This carried over to our estimate of the overall proportion of relevant for the R_2 set, and thus to the total utility. This raises the question of how much confidence we can have in our estimates of utility, and is the subject of the next section.

7 How Accurate are Our Estimates of Utility?

The pooling and stratified sampling approaches are based on judging only a subset of each submitted set, so in neither case will the estimates of utility be perfect. A common measure of the distance between an estimate, $\hat{\mu}$, and the quantity we want to estimate, μ , is the mean square error (MSE) [1, p. 15]. The MSE of an estimate is the expected value of the square of the difference between the estimate and the true value:

$$\text{MSE}[\hat{\mu}] = E[(\hat{\mu} - \mu)^2] \quad (23)$$

Letting $m = E[\hat{\mu}]$, the MSE can be rewritten as the sum of two terms:

$$\begin{aligned} \text{MSE}[\hat{\mu}] &= E[(\hat{\mu} - \mu)^2] \\ &= E[((\hat{\mu} - m) - (\mu - m))^2] \\ &= E[(\hat{\mu} - m)^2 - 2(\mu - m)(\hat{\mu} - m) + (\mu - m)^2] \\ &= E[(\hat{\mu} - m)^2] - E[2(\mu - m)(\hat{\mu} - m)] + E[(\mu - m)^2] \\ &= E[(\hat{\mu} - m)^2] - 2 \times 0 \times E[(\hat{\mu} - m)] + (\mu - m)^2 \\ &= E[(\hat{\mu} - m)^2] + (\mu - m)^2 \\ &= E[(\hat{\mu} - E[\hat{\mu}])^2] + (\mu - E[\hat{\mu}])^2 \\ &= \text{Var}[\hat{\mu}] + \text{Bias}[\hat{\mu}]. \end{aligned}$$

The first term:

$$\text{Var}[\hat{\mu}] = E[(\hat{\mu} - E[\hat{\mu}])^2]$$

is the *variance* of the estimator $\hat{\mu}$ and measures the tendency of the estimator to deviate from its own expected value. The second term:

$$\text{Bias}[\hat{\mu}] = (E[\hat{\mu}] - \mu)^2$$

is the *bias* of $\hat{\mu}$ and measures the systematic difference between the expected value of the estimator and the value we are trying to estimate. It is often, though not always, desirable to use *unbiased* estimates of a quantity. An estimate is unbiased if $E[\hat{\mu}] = \mu$, i.e. $\text{Bias}[\hat{\mu}] = 0$.

These two concepts, bias and variance, and their sum the MSE, will be useful in discussing the accuracy of our estimates of utility.

7.1 Accuracy of Pooled Estimates

The MSE of the pooled estimates is difficult to determine, since the pool is not constructed randomly. The variance of a pooled estimate is nonzero, since we do not sample the entire population. However, the variance is likely to be smaller than that of the corresponding stratified estimate, due to the large number of documents judged.

The pooled estimate also has a nonzero bias, since if there are any relevant documents in the submitted set which were not judged, the estimated utility will be lower than the true utility. In fact, not only is the expected value of the pooled estimate always less than or equal to the true utility, but the actual value of the pooled estimate is always less than or equal to the true utility. So the pooled estimate is a lower bound on the true utility.

7.2 Accuracy of Estimates Based on Random Sampling

Recall that the utility of a submitted set can be expressed in terms of the proportion of relevant documents in that set:

$$u_i = ((u_{ai} - u_{bi})p_i + u_{bi})N_i \quad (24)$$

Similarly, we can estimate the utility of a submitted set based on an estimate of the proportion of relevant documents in that set:

$$\hat{u}_i = ((u_{ai} - u_{bi})\hat{p}_i + u_{bi})N_i \quad (25)$$

The MSE of such an estimate is

$$\begin{aligned} \text{MSE}[\hat{u}_i] &= E[(\hat{u}_i - u_i)^2] \\ &= E[((u_{ai} - u_{bi})\hat{p}_i + u_{bi})N_i - ((u_{ai} - u_{bi})p_i + u_{bi})N_i]^2 \\ &= E[(u_{ai} - u_{bi})^2 N_i^2 (\hat{p}_i - p_i)^2] \\ &= (u_{ai} - u_{bi})^2 N_i^2 E[(\hat{p}_i - p_i)^2] \\ &= (u_{ai} - u_{bi})^2 N_i^2 \text{MSE}[\hat{p}_i]. \end{aligned} \quad (26)$$

So the MSE of \hat{u}_i is a simple function of the MSE of our estimate of the proportion of relevant documents. For simple random sampling and stratified sampling, the estimates of the proportion are unbiased, that is $E[\hat{p}_i] = p_i$. Therefore, the MSE of \hat{p}_i results solely from its variance, and we have:

$$\text{MSE}[\hat{u}_i] = (u_{ai} - u_{bi})^2 N_i^2 \text{Var}[\hat{p}_i]. \quad (27)$$

Also note that \hat{u}_i is unbiased as well, so its MSE consists solely of variance.

In the rest of this section we will look at what \hat{p}_i 's variance is under different sampling techniques.

7.2.1 Variance of Proportions Estimated by Simple Random Sampling

We begin with the estimate produced by simple random sampling, as this is both a component of, and a point of comparison with, the stratified sampling method used for the filtering evaluation. Recall that our estimator of the proportion of relevant documents, based on a simple random sample from a set, is:

$$\hat{p} = \frac{a}{n} \quad (28)$$

where n is the size of the simple random sample, and a is the number of relevant documents in the sample. We cannot know the exact variance of this estimate without knowing the actual value of p , which is of course what we are trying to estimate in the first place. However, an unbiased estimate of the variance of our estimate of the proportion is [1, p. 52]:

$$\begin{aligned} \widehat{\text{Var}}[\hat{p}] &= \frac{N-n}{(n-1)N} \times \hat{p} \times (1-\hat{p}) \\ &= \frac{N-n}{(n-1)N} \times \frac{a}{n} \times \frac{n-a}{n} \\ &= \frac{(N-n)a(n-a)}{n^2(n-1)N} \end{aligned} \quad (29)$$

Suppose we used a simple random sample of size n_i from set R_i to estimate the utility of set R_i . Then the MSE of the resulting utility estimate for set R_i would have been:

$$\begin{aligned} \text{MSE}[\hat{u}_i] &= (u_{ai} - u_{bi})^2 N_i^2 \frac{(N_i - n_i)a_i(n_i - a_i)}{n_i^2(n_i - 1)N_i} \\ &= (u_{ai} - u_{bi})^2 N_i \frac{(N_i - n_i)a_i(n_i - a_i)}{n_i^2(n_i - 1)} \end{aligned} \quad (30)$$

NOTE: The *TREC-4* version of this paper had, instead of the above, the incorrect equation:

$$\text{MSE}[\hat{u}_i] = (u_{ai} - u_{bi})^2 \frac{(N_i - n_i)a_i(n_i - a_i)}{n_i^2(n_i - 1)N_i}$$

7.2.2 Variance of Proportions Estimated by Stratified Sampling

In stratified sampling we separately estimate the proportion of relevant in each stratum and combine these estimates to get an estimate of the overall proportion:

$$\hat{p}_i = \sum_{h \in R_i} \frac{N_h}{N_i} \hat{p}_h \quad (31)$$

By the properties of the variance of linear combinations of random variables, and the fact that our samples from the strata are independent, we have [1, p. 92]:

$$\text{Var}[\hat{p}_i] = \sum_{h \in R_i} \frac{N_h^2}{N_i^2} \text{Var}[\hat{p}_h] \quad (32)$$

Each \hat{p}_h is an estimate of the proportion of relevant in a stratum, based on a simple random sample from the stratum. Therefore, the results of the previous section tell us that an unbiased estimate of the variance of \hat{p}_h is:

$$\widehat{\text{Var}}[\hat{p}_h] = \frac{(N_h - n_h)a_h(n_h - a_h)}{n_h^2(n_h - 1)N_h} \quad (33)$$

Substituting Equation 33 into Equation 32 then gives us an unbiased estimate of the variance of our stratified estimate of the proportion of relevant in the R_i set:

$$\begin{aligned} \widehat{\text{Var}}[\hat{p}_i] &= \sum_{h \in R_i} \frac{N_h^2}{N_i^2} \frac{(N_h - n_h)a_h(n_h - a_h)}{n_h^2(n_h - 1)N_h} \\ &= \frac{1}{N_i^2} \sum_{h \in R_i} \frac{N_h(N_h - n_h)a_h(n_h - a_h)}{n_h^2(n_h - 1)} \end{aligned} \quad (34)$$

Further substituting Equation 34 into Equation 27 gives us the MSE for the estimate \hat{u}_i (Equation 25) based on the stratified estimate of \hat{p}_i :

$$\begin{aligned} \text{MSE}[\hat{u}_i] &= (u_{ai} - u_{bi})^2 N_i^2 \widehat{\text{Var}}[\hat{p}_i] \\ &= (u_{ai} - u_{bi})^2 N_i^2 \frac{1}{N_i^2} \sum_{h \in R_i} \frac{N_h(N_h - n_h)a_h(n_h - a_h)}{n_h^2(n_h - 1)} \\ &= (u_{ai} - u_{bi})^2 \sum_{h \in R_i} \frac{N_h(N_h - n_h)a_h(n_h - a_h)}{n_h^2(n_h - 1)} \end{aligned} \quad (35)$$

If we compare Equation 35 to Equation 30, we see that the stratified estimate has a smaller MSE than an estimate based on simple random sampling when:

$$\sum_{h \in R_i} \frac{N_h(N_h - n_h)a_h(n_h - a_h)}{n_h^2(n_h - 1)} < \frac{N(N - n)a(n - a)}{n^2(n - 1)}$$

This is almost always true when reasonable strata are defined and appropriately sized samples are chosen from those strata [1, p. 99].

NOTE: The *TREC-4* version of this paper had, instead of the above, the incorrect inequality:

$$\sum_{h \in R_i} \frac{N_h(N_h - n_h)a_h(n_h - a_h)}{n_h^2(n_h - 1)} < \frac{(N - n)a(n - a)}{n^2(n - 1)N}$$

8 Stratified Sampling: An Example (Part II)

Returning to our example, we can use Equation 35 to give the MSE's of the utility estimates in Equations 20–22:

$$\text{MSE}[\hat{u}_1] = 4^2(0 + 0 + 0 + \frac{64400}{26100}) = 39.5 \quad (36)$$

$$\text{MSE}[\hat{u}_2] = 2^2(0 + \frac{6800000}{26100} + 0 + \frac{64400}{26100}) = 1052.0 \quad (37)$$

$$\text{MSE}[\hat{u}_3] = 4^2(\frac{28130000}{26100} + \frac{6800000}{26100} + 0 + \frac{64400}{26100}) = 21452.5 \quad (38)$$

Recall that the \hat{u}_i are unbiased, so the MSE of each estimate is just its variance, i.e. $\text{MSE}[\hat{u}_1] = \text{Var}[\hat{u}_1]$. Making the usually reasonable assumption that \hat{u}_i has a roughly normal distribution, then a 95% confidence interval around \hat{u}_i is [12, ch. 7]:

$$\hat{u}_i \pm 1.96\sqrt{\text{Var}[\hat{u}_i]}.$$

Then combining Equations 20–22 with Equations 36–38, and using the above expression for the confidence interval gives:

$$\begin{aligned} u_1 &= 2.672 \pm 12.3 \\ u_2 &= -51.35 \pm 63.6 \\ u_3 &= -717.2 \pm 287.1 \end{aligned}$$

We of course arranged this example so that the true utilities (Equations 17 to 19), which are known in our example but which would not be known in general, fell within the 95% confidence intervals. This would usually be the case in practice.

9 TREC-5 Results

The results of the TREC-5 filtering evaluation appear in an Appendix to these proceedings. The seven sites that participated in the filtering evaluation, and the code names for their ten filtering system submissions were City Univ. (*city96f*), ITI (*iti96f*), Intext (*INTXA* and *INTXM*), U Mass (*INR3*), U Illinois (*ispF*), Queens (*pircs96f*), and Xerox (*xerox.f1*, *xerox.f2*, and *xerox.f3*). In the Appendix, Table 1 for each site shows the raw data used in computing utility estimates. For each of the forty-nine topics and each of the three runs, we see the number of documents submitted and the pooled and stratified estimates of the utility of those submitted sets. Additional tables provide both summary and graphical presentations of this data.

Harman's summary of TREC-5 elsewhere in these proceedings discusses the nature of the topic set, training set, and test set used for the routing and filtering evaluations. From the standpoint of the filtering evaluation, the most notable characteristic of the data was that the distribution of relevant documents was more skewed than in TREC-4. There were more topics with few or no relevant documents found in the test set, and a few topics with a very large number of relevant documents. This skewing, combined with the variety of training data sets, most of unclear relation to the test data, made the filtering task quite challenging.

In the rest of this section we make a few observations on the approaches taken by TREC-5 filtering sites, briefly consider the effectiveness of various systems, and discuss how the properties of the TREC-5 data affected the methods used to estimate utility.

9.1 Approaches

As mentioned earlier, the TREC-5 filtering track used the same training and test data as the TREC-5 routing evaluation. Six of the seven filtering sites took advantage of this by basing their filtering runs on one of their routing runs. These sites produced routing queries by whatever means and used them to generate a score (implicitly or explicitly) for each of the test documents. The top 1000 scoring test documents were submitted (sorted by scores) for the routing evaluation, while some other processing of the scores was used to choose the test documents submitted in the site's filtering runs. The exception to this general strategy was Intext, which did not take part in the routing evaluation. However, it too computed numeric scores for all test documents. No purely boolean or other non-numeric methods for filtering were tried at TREC-5.

All sites except U Illinois applied machine learning techniques to the known relevant and non-relevant training documents in an attempt to produce a query better than the original routing topic description. Most of the algorithms used gave more weight to the training documents than to the original topic description, a reasonable strategy given the large amount of training data available.

The machine learning strategies used varied widely, and details can be found in the individual sites' papers. One point of interest was that three sites (City, Queens, and U Mass) explicitly tuned their routing queries to optimize average precision, a measure of ranking effectiveness that is the focus of the routing evaluation. Despite this emphasis on the routing evaluation, these sites also turned in three of the four best performances on the filtering evaluation.

While simply computing scores for documents was sufficient to rank them, additional processing was necessary to produce the submitted sets of documents required for filtering. Four sites (City, ITI, Queens, and U Mass) used the training data to set a threshold on their raw scores. Each routing query was applied to some or all of the training documents, and the documents were sorted by the resulting score. The sorted list was then scanned to find the score that, if used as a threshold, optimized the appropriate utility measure on the training documents. This procedure was repeated for each of the three utility measures to find a threshold for each of the filtering runs. The query was then used to score test documents, and all test documents exceeding the appropriate threshold went into the submitted set for that filtering run.

Intext and Xerox also generated their submitted sets by thresholding, but chose thresholds in different fashion. Xerox's logistic regression approach produces raw scores which are estimates of the probability of relevance of a document. The Xerox runs assumed these probability estimates were in fact correct, in which case the optimal thresholds follow from the definition of the utility measures (see Section 4.5). Intext found the highest scoring training document for each of their filtering queries and set thresholds at fixed percentages of this score. Finally, U Illinois did not use thresholding at all, but simply submitted a fixed number of documents for each utility measure, regardless of the topic.

9.2 Effectiveness

We leave the analysis of TREC-5 filtering results largely up to the track participants, and make only a few observations here. Table 1 shows the mean rank of each of the 10 systems for the three runs and two effectiveness estimates. The ordering of the systems by mean rank is for the most

System	Run 1 (thr. 0.75)		Run 2 (thr. 0.5)		Run 3 (thr. 0.25)		Best Routing Ave. Precision
	Pooled	Strat	Pooled	Strat	Pooled	Strat	
INR3	3.61	3.49	3.55	3.31	2.71	2.84	0.3359
city96f	3.71	3.53	3.02	3.08	3.16	3.35	0.3475
xerox.f1	3.76	3.59	3.55	3.61	4.35	4.39	0.1223
pircs96f	4.10	4.12	3.24	3.33	3.88	4.02	0.3402
xerox.f2	4.14	4.10	3.59	3.71	4.12	4.29	0.1223
xerox.f3	5.14	5.14	4.78	4.92	5.02	5.16	0.1223
iti96f	6.37	6.45	6.82	6.86	7.14	6.65	0.1657
INTXA	6.53	6.59	6.92	6.92	6.80	6.65	-
INTXM	7.55	7.69	8.61	8.37	7.67	7.43	-
ispF	8.88	9.10	9.45	9.45	9.24	9.35	0.0196

Table 1: Mean rank of the 10 filtering systems under 6 evaluation conditions produced by pairing 3 runs with 2 estimation methods. (We show the optimal threshold on probability of relevance/instantaneous precision p for each run.) For each condition, the ten systems were sorted by utility and given a rank between 1 and 10 for each of the 49 topics. (If a group of systems had identical utilities for a topic their ranks were replaced by the mean rank for the group.) The mean of those ranks, over the 49 topics, is shown in this table. Systems are sorted by the mean over the resulting six means. The last column shows that the best average precision achieved by the site in the routing evaluation is, except for Xerox, roughly correlated with the effectiveness of the site's filtering runs.

part consistent across runs and effectiveness measures. The last column of Table 1 shows the best average precision (over 45 topics) for the routing runs submitted by each filtering site that did the routing task. (We did not attempt to establish a correspondence between particular routing submissions and particular filtering submissions.) Roughly speaking, the sites that did well at routing did well at filtering, and vice versa. Xerox was a bit of an exception to this, raising the interesting possibility that probability estimates that aren't good enough for ranking may be good enough to do reasonable binary classification.

9.3 Estimating Utilities

The widely varying number of relevant documents for different filtering topics stressed our estimation methods in interesting ways, as reflected in the relatively large number of cases where the pooled estimate of utility falls outside the 95% confidence interval for the corresponding stratified estimate of utility. In Table 2, we show the number of times the pooled estimate is above or below the outer limit of confidence interval on the stratified estimate. (The raw data is taken from Table 1 for each site in the filtering Appendix.)

These large disagreements between the pooled and stratified estimates occur in two very different situations. First, there are 48 cases where the pooled estimate is higher than the upper end of the confidence interval for the stratified estimate. At first glance, this seems quite surprising, since the pooled estimate is guaranteed to be an *underestimate* of the true utility, while the stratified estimate is an unbiased estimate of true utility.

The fault here is actually in the confidence interval estimation. As the parenthesized values in Table 2 show, in 32 of these 48 cases the estimated standard deviation is 0, so the confidence interval degenerates to a point. One way we can get a degenerate confidence interval is when the submitted set has 100 or fewer items. In this case, all documents in the set are judged and there is no sampling error. However, in this case the stratified and pooled estimates will always agree exactly.

The other, pernicious, way we can get a degenerate confidence interval is when no relevant documents are found in the submitted set. This leads to a statistically unbiased, but usually wrong, estimate that the sampling error is 0. Table 3 shows that when the pooled estimate is higher than the top of confidence interval, the submitted set tends to be large (median size of 256.0), and the number of relevant in the submitted set (as estimated by pooling) tends to be small (median 6.5). This is the worst case for our sampling method: attempting to estimate a small value (the proportion of relevant) by drawing a small sample from a large set.

In this situation, the likely result is that we will see no relevant documents in some or all strata. When no relevant documents are found in any strata we get a degenerate confidence interval, as seen in the 32 cases mentioned above. While we have not analyzed the remaining 16 cases, it is likely that many of them result from finding relevant documents in some strata but not others. This would result in an estimated standard deviation which is nonzero, but usually too low. In other words, our stratified estimate is less accurate than it appears. The pooled estimate, which uses a much larger sample size, is likely to be more accurate. This is likely to be true for some cases where the pooled estimate is within the confidence interval for the stratified estimate as well.

On the other hand, the 62 cases where the pooled estimates are lower than the bottom of the

	Run 1 (thr. 0.75)	Run 2 (thr. 0.5)	Run 3 (thr. 0.25)	Totals
$\hat{u}_{i,p} < \hat{u}_i - 1.96\sqrt{\text{Var}[\hat{u}_{i,s}]}$	11 (0)	20 (0)	31 (0)	62 (0)
$\hat{u}_i - 1.96\sqrt{\text{Var}[\hat{u}_{i,s}]} \leq \hat{u}_{i,p} < \hat{u}_{i,s}$	17 (0)	49 (0)	56 (0)	122 (0)
$\hat{u}_{i,s} = \hat{u}_{i,p} = \hat{u}_i$	447 (447)	364 (364)	344 (344)	1155 (1155)
$\hat{u}_{i,s} < \hat{u}_{i,p} \leq \hat{u}_i + 1.96\sqrt{\text{Var}[\hat{u}_{i,s}]}$	13 (0)	33 (0)	37 (0)	83 (0)
$\hat{u}_i + 1.96\sqrt{\text{Var}[\hat{u}_{i,s}]} < \hat{u}_{i,p}$	2 (2)	24 (16)	22 (14)	48 (32)
Totals	490 (449)	490 (380)	490 (358)	1470 (1187)

Table 2: Comparison of pooled estimate of utility, $\hat{u}_{i,p}$, with stratified estimate of utility, $\hat{u}_{i,s}$ for filtering submissions. There are 10 systems, 3 runs (column header shows threshold on probability of relevance for run), and 49 topics for a total of 1470 pairs of estimates. We break this total down by runs and by the relationship of $\hat{u}_{i,p}$ to $\hat{u}_{i,s}$ and to the bounds of a 95% confidence interval on $\hat{u}_{i,s}$. In parentheses we show the number of times the confidence interval on $\hat{u}_{i,s}$ degenerates to a point.

confidence interval on the stratified estimates exhibit a strength of the stratified approach. As Table 3 shows, the submitted sets are typically large here also (median size 562.0), but so is the number of relevant in the submitted set (median value 282.5 as estimated by pooling). These 62 cases involve only 7 topics, which are 7 of the 9 topics with the highest number of known relevant documents. (In fact, 58 of the cases occur on the four topics where the pooled sample reveals there are at least 500 relevant documents.) This is a situation where the stratified estimates of both utility and sample variance will be quite accurate. In contrast, the pooling assumption (that unjudged documents are non-relevant) is most likely to be wrong in these cases. For instance, our stratified sample lets us say with high confidence that the set submitted by system *iti96f* for Run 3 on Topic 111 included 2254 ± 441.1 relevant documents. Since the entire judged pool for this topic contained only 1303 documents (887 of which were found to be relevant) the pooled estimate cannot help but substantially underestimate the true utility.

10 Future Filtering Tracks

The TREC-5 filtering track was an advance over the TREC-4 filtering track from the standpoint of planning (the evaluation procedures were finalized well before participants retrieved the data sets, unlike in TREC-4) and participation (seven sites participating vs. four sites for TREC-4). However, much remains to be improved. First, as pointed out in Section 4.5, all the filtering track runs required high precision by the standards of current IR systems. This fact, combined with the small number of relevant documents available for many topics, led to a situation where the optimal behavior for most systems on many topics was to submit a set of size 0. This is obviously not conducive to either comparing systems or understanding the behavior of a single system. In future filtering evaluations it will be desirable to adjust the data sets and/or the effectiveness measures to

	Median Submitted Set Size	Median Relevant in Submitted Set (pooled estimate)	Median Relevant in Pool
$\hat{u}_{i,p} < \hat{u}_i - 1.96\sqrt{\text{Var}[\hat{u}_{i,s}]}$	562.0	282.5	808.0
all runs	10.0	2.0	42.0
$\hat{u}_i + 1.96\sqrt{\text{Var}[\hat{u}_{i,s}]} < \hat{u}_{i,p}$	256.0	6.5	20.0

Table 3: Statistics on submitted sets where the pooled estimate of utility ($\hat{u}_{i,p}$) is outside the 95% confidence interval for the stratified estimate of utility ($\hat{u}_{i,s}$). We compare with corresponding values for the collection of all submitted sets for all topics, systems, and the three runs. We show median values for the size of the submitted set, the number of relevant in submitted set (estimated using the pooling assumption), and the number of relevant in the judged pool for the corresponding topic.

avoid this situation. In any case, utility-based measures make it difficult to compute the average effectiveness of systems across topics, and it is unclear how well they capture user needs. So non-utility measures need to be investigated.

Our estimation procedures could be improved as well. A more careful choice of sample sizes in stratified sampling could improve our estimates there. It might also be possible to increase the number of documents judged for filtering submissions, since there is substantial redundancy between them and routing submissions from the same sites.

Since most sites have chosen to base their filtering systems very closely on their routing systems, it seems worth investigating more closely the relationship between the two. For instance, Chris Buckley has suggested examining the full rankings produced by filtering systems to find the optimal score that *could* have been produced, as a measure of how well systems are setting their thresholds.

Finally, the TREC filtering and routing evaluations have been criticized as being unrealistic, particularly with respect to the training data supplied. On the one hand, the task is made too easy, by supplying more training data, with a higher density of relevant documents, than would be available in most real routing or binary classification applications. On the other hand, the task is too difficult, in that training data is drawn from sources that have little or no relation to the test data, and the procedures used to obtain the training data (pooling of previous years ad hoc runs) are complex and poorly understood. One goal for future evaluations will be to provide more realistic training and test data, ideally drawn from a chronologically ordered stream of documents.

11 Summary

The TREC-5 filtering track solidified the role of an evaluation of binary text classification in TREC. Seven sites participated in the track, producing data both on their effectiveness of their systems and on the appropriateness of evaluation strategies. We encourage all TREC participants to consider taking part in future TREC filtering evaluations.

12 Acknowledgments

I am greatly appreciative of Donna Harman, Ellen Voorhees, and the rest of the team at NIST, for both their work in conducting the filtering evaluation and their suggestions for improving it. The ideas presented here were developed in extensive discussions with the members of the TREC-1 through TREC-5 program committees (particularly Chris Buckley, David Hull, and Karen Sparck Jones), and with TREC-4 and TREC-5 participants (particularly David Hull, Paul Kantor, K. L. Kwok, and Julian Yochum).

References

- [1] William G. Cochran. *Sampling Techniques*. John Wiley & Sons, New York, 3rd edition, 1977.
- [2] William S. Cooper. On selecting a measure of retrieval effectiveness. *Journal of the American Society for Information Science*, 24:87–100, March–April 1973.
- [3] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. Wiley-Interscience, New York, 1973.
- [4] Donna Harman. Overview of the fourth Text REtrieval Conference (TREC-4). In D. K. Harman, editor, *The Fourth Text REtrieval Conference (TREC-4)*, Gaithersburg, MD, 1996. U. S. Dept. of Commerce, National Institute of Standards and Technology.
- [5] Marti Hearst, Jan Pedersen, Peter Pirolli, Hinrich Schütze, Gregory Grefenstette, and David Hull. Xerox site report: Four TREC-4 tracks. In D. K. Harman, editor, *The Fourth Text REtrieval Conference (TREC-4)*, pages 97–119, Gaithersburg, MD, 1996. U. S. Dept. of Commerce, National Institute of Standards and Technology.
- [6] Paul S. Jacobs. GE in TREC-2: Results of a boolean approximation method for routing and retrieval. In D. K. Harman, editor, *The Second Text Retrieval Conference (TREC-2)*, pages 191–199, Gaithersburg, MD, March 1994. U. S. Dept. of Commerce, National Institute of Standards and Technology. NIST Special Publication 500-215.
- [7] K. L. Kwok, L. Grunfeld, and D. D. Lewis. TREC-3 ad-hoc, routing retrieval and thresholding experiments using PIRCS. In D. K. Harman, editor, *Overview of the Third Text REtrieval Conference (TREC-3)*, pages 247–255, Gaithersburg, MD, April 1995. U. S. Dept. of Commerce, National Institute of Standards and Technology.
- [8] David D. Lewis. Evaluating text categorization. In *Proceedings of Speech and Natural Language Workshop*, pages 312–318. Defense Advanced Research Projects Agency, Morgan Kaufmann, February 1991.
- [9] David D. Lewis. Evaluating and optimizing autonomous text classification systems. In Edward A. Fox, Peter Ingwersen, and Raya Fidel, editors, *SIGIR '95: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 246–254, New York, 1995. Association for Computing Machinery.

- [10] David D. Lewis. The TREC-4 filtering track. In D. K. Harman, editor, *The Fourth Text REtrieval Conference (TREC-4)*, pages 165–180, Gaithersburg, MD, 1996. U. S. Dept. of Commerce, National Institute of Standards and Technology.
- [11] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In W. Bruce Croft and C. J. van Rijsbergen, editors, *SIGIR 94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12, London, 1994. Springer-Verlag.
- [12] David S. Moore and George P. McCabe. *Introduction to the Practice of Statistics*. W. H. Freeman, New York, 1989.
- [13] S. E. Robertson. The probability ranking principle in IR. *Journal of Documentation*, 33(4):294–304, December 1977.
- [14] Richard M. Tong and Lee A. Appelbaum. Machine learning for knowledge-based document routing (a report on the TREC-2 experiment). In D. K. Harman, editor, *The Second Text Retrieval Conference (TREC-2)*, pages 253–264, Gaithersburg, MD, March 1994. U. S. Dept. of Commerce, National Institute of Standards and Technology. NIST Special Publication 500-215.

NLP Track at TREC-5

Summarized by Tomek Strzalkowski from notes by Karen Sparck Jones and himself

ABSTRACT

NLP track has been organized for the first time at TREC-5 to provide a more focused look at how NLP techniques can help in achieving better performance in information retrieval. The intent was to see if NLP techniques available today are mature enough to have an impact on IR, specifically if and when they can offer an advantage over purely quantitative methods. This was also a place to try some more expensive and more risky solutions than those used in main TREC evaluations.

1. AIMS

More specifically, there were two principal aims of NLP track evaluations:

1. To see whether NLP has value in specific retrieval circumstances even if it has not hitherto been proven advantageous for routine document/text indexing and retrieval.
2. To see if NLP can be effectively used as a means to translate an NL text into whatever representation the search engine allows: this applies to either documents or queries, or both. In term-based systems, we have a representation that is basically: terms + weights + "=" (i.e., equivalence relation between terms). Can NLP help to get closer to the 'optimal' query.

2. PARTICIPANTS

Five teams participated in this NLP track: GE/Rutgers/NYU/Lockheed Martin, Xerox, Mitre, Claritech, and ISS Singapore. Results were submitted by the first four teams only. In addition, Chris Buckley supplied baselines for Sabir/SMART system. Other "baselines" were created by GE and Xerox teams running their system in no-NLP mode.

3. EVALUATION SETUP

The evaluation was done in the ad-hoc retrieval mode only. Both automatic and manual modes were allowed. In an automatic run, no human intervention was permitted at any stage. In a manual run, queries could be expanded or modified manually, by adding or deleting terms or text, including from any documents in the test collection.

4. RESULTS

All systems did better than SMART statistical baseline, some substantially so (see attached recall-precision graphs). At least three out of the four systems used some kind of phrase extraction

mechanism based on more or less elaborate syntactic analysis of text. This is worth noting particularly because the SMART baseline system extracts rudimentary statistical “phrases” (adjacent word bigrams) to expand word-only indexing. Thus, at least in this particular setup, linguistic phrases seem more effective than adjacency bigrams.

FIGURE 1. NLP Track Summary: Best Results

run id	GENLP4	CLARMC	xerox_nlp5	Mitre	SMART
type	manual	manual	manual	manual	auto. base
11pt prec %change	0.3176 +79	0.2842 +60	0.2320 +31	0.1896 +7	0.1771
R-prec. %change	0.3090 +70	0.2934 +61	0.2490 +37	0.1859 +2	0.1823

run id	xerox_nlp4	GENLP3	CLPHR1	SMART
type	automatic	automatic	automatic	auto. base
11pt prec %change	0.2280 +29	0.2220 +25	0.2010 +13	0.1771
R-prec. %change	0.2460 +35	0.2242 +23	0.2127 +17	0.1823

In addition to phrase-based indexing, full-text query expansion experiments performed by GE-led team showed very promising results. In this method, original search queries are expanded adding entire text passages from any documents containing related material. See Strzalkowski et al. paper for details.

Claritech team experimented with several alternative phrase extracting methods for document indexing. These included head-modifier pairs, adjacent subphrases, and full noun phrases. Phrases were obtained using very fast, shallow noun phrases parser. Further experiments included various combinations of phrase indexing methods and traditional single word indexing. Claritech results show the strongest gain from phrasal indexing. See Evans et al. paper for details.

GE/NYU/Rutgers/Lockheed Martin team used “stream-based” architecture to evaluate several phrase-indexing approaches, including head+modifier representation obtained via full syntactic parsing of entire data set. GE’s head+modifier pairs include verb+object and subject+verb combinations in addition to pairs obtained from noun phrases. Precision gains were less than for Clarit system, with unnormalized phrases slightly outperforming the more advanced head+modifier representation. In addition, manual and automatic full-text query expansion methods have been used, producing very encouraging results.

Mitre’s experiments were limited to using part-of-speech tagger and applying differential term weighting depending upon its part of speech. They noted only minimal gains over statistical SMART baseline. See Burger et al. paper for details.

Xerox group’s goal was to recreate on a larger scale Joel Fagan’s experiments in which he compared the effects of using syntactic and statistical phrases for document indexing. Statistical phrases were obtained using adjacent word pairs that occurred with certain frequencies in the data set. Syntactic phrases were derived with a “light-weight” phrasal parser, but no normalization (e.g., head-modifier) was performed. These experiments showed only very modest improvement over non-NLP baseline. For details please see Grefenstette et al. paper.

5. CONCLUSIONS

This NLP track demonstrated that natural language processing techniques have solid but limited impact on the quality of text retrieval, particularly precision. Techniques aimed at producing higher quality queries, e.g., query expansion, constraints, appear to be more effective than those aimed primarily at obtaining improved indexing of database documents. More work is needed before more substantial gains can be seen, including the use of more advanced, and therefore more expensive, semantic analysis techniques.

Figure 2 summarizes a rather subjective view of which NLP techniques have been tried in information retrieval, and what might be their potential for improving retrieval precision. This chart was discussed at the NLP track workshop on the last day of TREC-5 meeting. It was decided that NLP techniques that show particular promise in relatively smaller-scale track evaluations should be transferred to main evaluations as soon as practical.

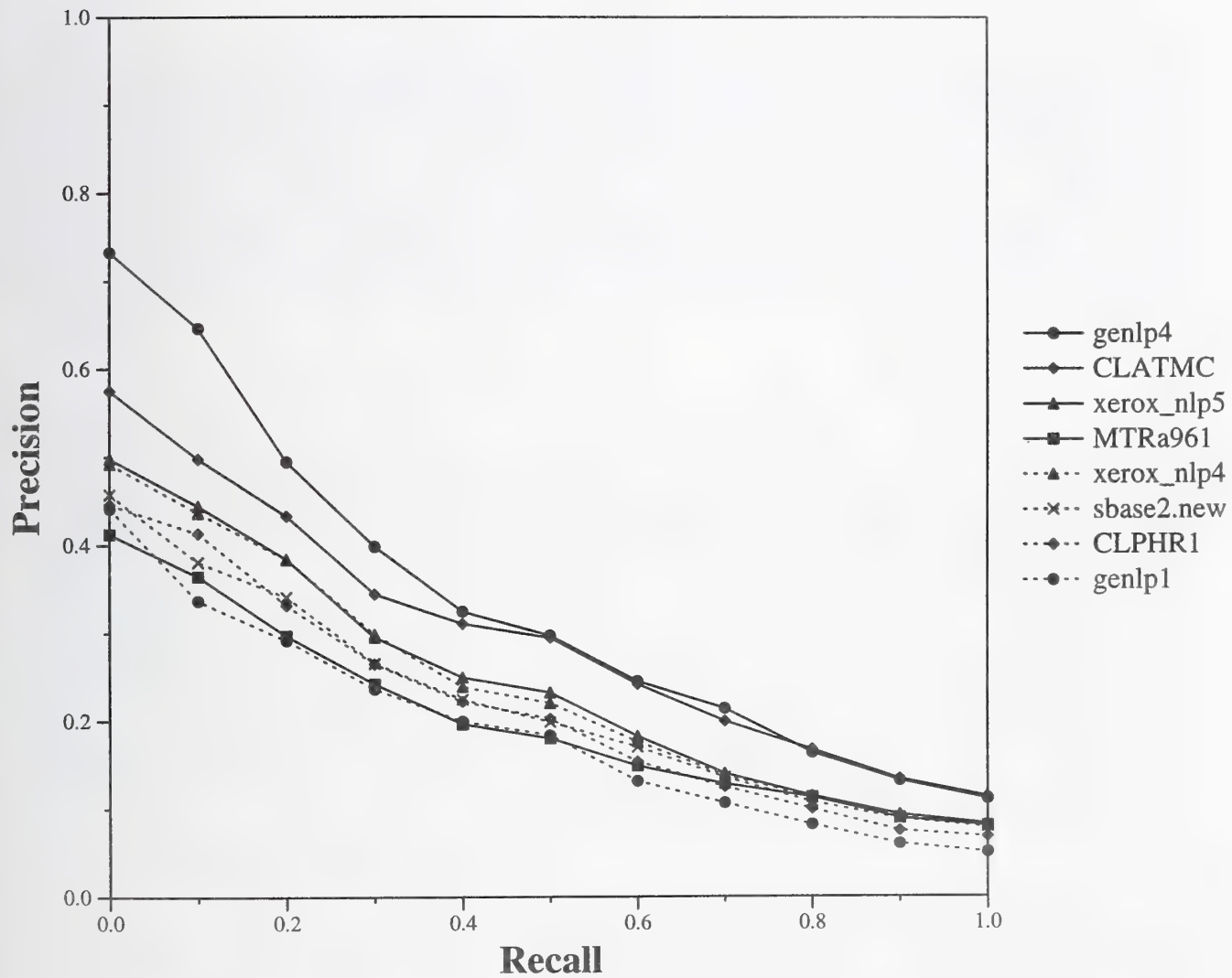
ACKNOWLEDGEMENTS

We would like to thank Donna Harman, Ellen Voorhees, and Dawn Hoffman, of NIST for their support in organizing of NLP track evaluations. This project was supported in part by the Defense Advanced Research Projects Agency under the Tipster Phase 2 and Phase 3 contracts 94-FI59900-000, and 97-FI56800-000.

FIGURE 2. NLP results analysis: a subjective view

NL technique	class	%change precision
Full-text query expansion	query build	40 to ???
Term-based query expansion	query build	15 to 25
deleting extraneous text from queries	query build	0 to 5
hyphenated phrases	phrases	-15
word bi-grams	phrases	5 to 10
extended bi-grams (windows)	phrases	-5
FSA phrases (noun groups)	phrases	7 to 25
Head+Modifier Pairs (full parsing)	phrases	2 to 15
proper names	concepts	1 to 3
concept tagging for indexing	concepts	0 to ???
concept tagging for re-ranking	concepts	0 to 3
stylistics	discourse	0 to ???
lexical normalization	stemming	5 to 8

BEST NLP



The TREC-5 Database Merging Track

Ellen M. Voorhees
National Institute of Standards and Technology
Gaithersburg, MD
ellen.voorhees@nist.gov

There are many times when users want to search separate text collections as if they were a single collection. For example, computer networks can provide access to a variety of corpora that are owned and maintained by different entities. Instead of issuing search commands to each of the databases in turn and manually collating the individual results, users prefer a mechanism for performing a single, integrated search. In other cases, reliability and efficiency concerns may dictate that databases that are under the same administrative control should be physically separate. Again, users want to issue a single search request that returns an integrated result. The database merging track investigates methods for combining the results of separate searches into a single, cohesive result.

1 The Task

The initial running of the database merging track occurred in TREC-4. To foster participation by allowing as many different types of merging strategies as possible, the task in the TREC-4 track was left very open: the data was split into ten collections (corresponding to each source on each TREC disk used in the ad hoc task) and participants were free to produce a merged result any way they saw fit.

The task in TREC-5 was somewhat more focussed. The track used the same topics as the ad hoc task (topics 251–300), and the same documents as the ad hoc task (the documents on TREC disks 2 and 4). (This allowed the track to contribute to the ad hoc relevance assessment pools, and to use those pools to evaluate the runs.) The documents on the two disks were partitioned into 98 different databases, with each partition containing documents from a single source.¹ Participants were required to produce a ranking of the documents for each topic *without* searching every database for every topic. That is, merging strategies that routinely search all available databases were specifically excluded from the track.

Track participants could submit up to two merging runs, and were required to submit a comparable ad hoc run (all documents in a single collection) to provide a baseline for comparisons.

Database merging consists of two sub-problems: *resource discovery* and *result combination*. Resource discovery is deciding which of the set of available databases should be searched for the current query; result combination is producing one ranked list of documents from the results of the sources searched. The decisions to significantly increase the number of databases over the TREC-4 task and to exclude methods that always search all databases were made to focus this running of the track on the resource discovery subproblem.

¹The databases were defined by a script created by the group at the University of Massachusetts at Amherst. Contact Ellen Voorhees at NIST (ellen.voorhees@nist.gov) for a copy. Category B participants used WSJ90, WSJ91, and WSJ92 as separate databases.

2 Participants

Three groups participated in the TREC-5 track. See the respective papers by these groups elsewhere in the proceedings for more details regarding their results.

Université de Neuchâtel: The Université de Neuchâtel group used TREC-5 to investigate a retrieval model based on logistic regression that treats data fusion (combining different search schemes) and database merging (combining distributed information services) to be different facets of the same problem. They submitted two category B database merging runs using both long and short topics (*UniNE0* and *UniNe9*).

Australian National University: This group used the resource discovery emphasis of the track to examine the specific problem of selecting network servers. In addition to retrieval effectiveness, their work examined the efficiency measure of the number of servers that needed to be contacted to produce the result. They submitted three category A runs: *anu5mrg0*, their baseline ad hoc run; *anu5mrg1*, a run that used historical data to pick servers; and *anu5mrg7*, a run that used lightweight probes to pick servers.

FS Consulting: This group used their database merging track entry to measure the effectiveness of their document scoring algorithms when searching across multiple databases. They found the document scoring algorithm to be stable for widely varying numbers of databases. FS Consulting submitted one category A database merging run, *fsclt3m*, which is comparable to their ad hoc submission, *fsclt3*.

3 Future of the Track

Unfortunately, the database merging track has proven to be a high overhead track for participants. Despite generally high interest in the problem addressed by the track, the track has attracted few participants, likely because of the amount of data manipulation it requires as compared to other TREC tracks. The track will be suspended for at least a year while a simpler track design is sought.

Using Query Zoning and Correlation Within SMART : TREC 5

Chris Buckley*, Amit Singhal, Mandar Mitra

Abstract

The Smart information retrieval project emphasizes completely automatic approaches to the understanding and retrieval of large quantities of text. We continue our work in TREC 5, performing runs in the routing, ad-hoc, and foreign language environments. The major focus this year is on “zoning” different parts of an initial retrieval ranking, and treating each type of query zone differently as processing continues. We also experiment with dynamic phrasing, seeing which words co-occur with original query words in documents judged relevant. Exactly the same procedure is used for foreign language environments as for English; our tenet is that good information retrieval techniques are more powerful than linguistic knowledge.

Introduction

For over 30 years, the Smart project at Cornell University, under the direction of the late Gerry Salton, has been investigating the analysis, search, and retrieval of heterogeneous text databases, where the vocabulary is allowed to vary widely, and the subject matter is unrestricted. Our belief is that text analysis and retrieval must necessarily be based primarily on a study of the available texts themselves. The community does not understand natural language well enough at the present time to make use of a more complex text analysis. Knowledge bases covering the detailed structure of particular subject areas, together with inference rules designed to derive relationships between the relevant concepts, are very difficult to construct, and have not yet been proven to aid in general retrieval.

Fortunately very large text databases are now available in machine-readable form, and a substantial amount of information is automatically derivable about the occurrence properties of words and expressions in natural-language texts, and about the contexts in which the words are used. This information can help in determining whether a query and a text are semantically homogeneous, that is, whether they cover similar subject areas. When that is the case, the text can be retrieved in response to the query.

Automatic Indexing

In the Smart system, the vector-processing model of retrieval is used to transform both the available information requests as well as the stored documents into vectors of the form:

$$D_i = (w_{i1}, w_{i2}, \dots, w_{it})$$

where D_i represents a document (or query) text and w_{ik} is the weight of term T_k in document D_i . A weight of zero is used for terms that are absent from a particular document, and positive weights characterize terms actually assigned. The assumption is that t terms in all are available for the representation of the information.

The basic “tf*idf” weighting schemes used within SMART have been discussed many times. For TREC 5 we use the same basic weights and document length normalization as were developed at Cornell by Amit

*Department of Computer Science, Cornell University, Ithaca, NY 14853-7501. This study was supported in part by the National Science Foundation under grant IRI 93-00124.

Singhal for TREC 4. Tests on various collections show that this indexing is reasonably collection independent and thus should be valid across a wide range of new collections. No human expertise in the subject matter is required for either the initial collection creation, or the actual query formulation.

The same phrase strategy (and phrases) used in all previous TRECs ([4, 2, 5, 6]) are used for TREC 5. Any pair of adjacent non-stopwords is regarded as a potential phrase. The final list of phrases is composed of those pairs of words occurring in 25 or more documents of the initial TREC 1 document set. Phrases are weighted with the same scheme as single terms.

Text Similarity Computation

When the text of document D_i is represented by a vector of the form $(d_{i1}, d_{i2}, \dots, d_{it})$ and query Q_j by the vector $(q_{j1}, q_{j2}, \dots, q_{jt})$, a similarity (S) computation between the two items can conveniently be obtained as the inner product between corresponding weighted term vectors as follows:

$$S(D_i, Q_j) = \sum_{k=1}^t (d_{ik} * q_{jk}) \quad (1)$$

Thus, the similarity between two texts (whether query or document) depends on the weights of coinciding terms in the two vectors.

System Description

The Cornell TREC experiments use the SMART Information Retrieval System, Version 12, and most were run on a dedicated Sun Sparc 20/51 with 160 Megabytes of memory and 33 Gigabytes of local disk (some supporting runs were made on a Sun UltraSparc 1/140).

SMART Version 12 is the latest in a long line of experimental information retrieval systems, dating back over 30 years, developed under the guidance of G. Salton. The new version is approximately 44,000 lines of C code and documentation.

SMART Version 12 offers a basic framework for investigations of the vector space and related models of information retrieval. Documents are fully automatically indexed, with each document representation being a weighted vector of concepts, the weight indicating the importance of a concept to that particular document (as described above). The document representatives are stored on disk as an inverted file. Natural language queries undergo the same indexing process. The query representative vector is then compared with the indexed document representatives to arrive at a similarity (equation (1)), and the documents are then fully ranked by similarity.

SMART is highly flexible and very fast, thus providing an ideal platform for information retrieval experimentation. Documents for TREC 5 are indexed at a rate of over a Gigabyte an hour, on hardware costing under \$10,000 new. Retrieval speed is similarly fast, with basic simple searches taking much less than a second a query.

Routing Methodology

For the past two TRECs we have concentrated on the ad-hoc tasks. In TREC 3 we worked on expansion of ad-hoc queries using techniques from relevance feedback. In TREC 4 we re-examined basic weighting approaches, particularly looking at document length normalization as a source of improvement.

For TREC 5 this year, we shift back to a focus on the routing task, with the hope that lessons learned there will help improve our results on the ad-hoc task. Our changes centered in four areas, each of which will be discussed:

- Adjust our routing parameters to the more accurate weights developed in the TREC 4 ad-hoc task.

- Explore the concept of a “query zone”. The properties of non-relevant documents somehow related to the query (not having low similarity) are different from those of the general non-relevant documents.
- Re-examine co-occurrence of pairs of terms to see if pairs commonly occurring in the relevant documents can improve retrieval.
- Explicitly use the correlation of pairs of terms to weight pairs added by a co-occurrence examination.

In TREC 4, we started with a basic Rocchio feedback approach, which modified the indexed weights of query terms based on the presence of terms in the judged relevant and non-relevant documents.

$$\begin{aligned}
 Q_{\text{new}} &= A * Q_{\text{old}} \\
 &+ B * \text{average_wt_in_rel_docs} \\
 &- C * \text{average_wt_nonrel_docs}
 \end{aligned}$$

The Rocchio parameters A, B, C had values 64,64,2 in TREC 4. We expanded by 50 single terms and 10 phrases that occurred in the relevant documents. We then modified these Rocchio weights by undergoing a six pass Dynamic Feedback Optimization (DFO) stage[3].

The original query weight was heavily emphasized in TREC 4, while the weight among the non-relevant documents was almost negligible. However, given the increased accuracy of our new weighting schemes, we can rely much more on the weights in the documents. For TREC 5, we use parameters 8,64,64 which are dominated by the weights in the documents as opposed to the original query. We also double the number of single terms being added to 100. The number of phrases is kept at 10, though additional pairs are added as described below. A final tweak is to only allow terms to be added if they occur in a certain percentage of the relevant documents. This prevents the randomly occurring terms from entering the query. For TREC 5 this randomness-threshold is set to 10% for single terms and 5% for phrases.

In a vector space model of information retrieval such as that used by SMART, a “query zone” can be thought of as a volume of the vector space which surrounds the query. It can be a tight query zone, consisting of those document vectors strongly related to the query vector, or a weak zone, composed of documents that have some undetermined relationship with the query. In either case, the properties of those documents in a “query zone” will be different from the properties of the vast majority of the documents in a large collection.

One use of a query zone is to define a set of documents that are hopefully within the same domain of the query, but not relevant to the query. The basic original query can easily distinguish a target set in which most relevant documents will appear; but the efforts to do so may interfere with the efforts to distinguish relevant from non-relevant documents within the target set. For example, the term “computer” may be a very good term for distinguishing the domain for a query about ‘which disk drive should I get for my Mac’, but it is not going to be useful within the technical domain. If the term is weighted too heavily (e.g., if it occurs in nearly all the relevant documents, but only 5% of the much larger collection of non-relevant documents), general articles about computers may get ranked too highly.

The query zone idea comes from Xerox’s “local region”. Schutze et al[7] used query specific screening to define a local region of the top 2000 documents for routing queries. Their motivation was to both reduce the number of documents to be used in their learning algorithms, and to learn to distinguish between the relevant documents and just the top non-relevant documents. While in some sense we are doing the latter, it is for different reasons (attempting to distinguish between the very top relevant and non-relevant documents with a single query is not generally successful) with a different final result. They end up with two queries and two thresholds to be applied to an incoming document; our goal is to come up with a single query.

For TREC 5, we use a query zone of 5000 documents to calculate the Rocchio weights (actually we use 5000 non-relevant documents, plus the judged relevant documents). This doesn’t affect the weights due to occurrences in the relevant documents, but the average weight of terms in the non-relevant documents is changed dramatically. Terms that serve only to identify the domain of the query will get substantially downweighted by this process.

This process can also have a positive effect. A term can be a relatively common term in the entire collection but can be a good term for distinguishing relevant documents from the non-relevant documents

in the query zone. An example would be a term like *tire* for the query *recycling of tires*. Such a term will get a higher Rocchio weight when weights are learned in the query zone in place of the entire collection.

In the routing environment this implies the system either keeps around the top 5000 indexed documents retrieved for each query, or does a new retrieval upon a retrospective collection for each invocation of the user routing profile. Either is somewhat expensive, but neither option is prohibitive.

The next area for exploration in routing is co-occurrence of terms. This has been explored by Cornell and other groups on and off for at least 20 years, most notably with the probabilistic dependency models, but never with great success. The object is to add pairs of terms that co-occur comparatively more often in the relevant documents than the non-relevant. Since the addition of pair information is primarily a high precision device, we decided to focus on a very narrow query zone. We consider co-occurrences among the relevant documents plus the top 2R non-relevant documents (where R is the number of relevant documents). Candidate pairs are those pairs of terms with at least one term occurring in the original query, and the other term being either a query term or an expansion term.

The weight of the candidate pair is given by the Rocchio formula, with the weight in a relevant or non-relevant document defined to be the minimum of the weights of the two component terms in that document. (Document weights at this point in the processing are defined without idf; idf is added later, and for pairs is the actual idf of the pair in the collection.) Like the single terms and phrases, pairs to be considered must meet a minimum occurrence threshold. For TREC 5 this was set to 7%; i.e., all candidate pairs must occur in at least 7% of the relevant documents.

Another option for defining the weight of a candidate pair is looked at in the second of our TREC 5 official runs. Instead of using the Rocchio formula, the pair weight is defined by the correlations of the pair and involved terms to relevance.

$$\begin{aligned} \text{PairWt} = & \text{Correlation_of_Pair_to_relevance} \\ & * (\text{Correlation_of_T}_1\text{_to_T}_2\text{_in_relevant_docs} \\ & - \text{Correlation_of_T}_1\text{_to_T}_2\text{_innreldocs}) \end{aligned}$$

An ideal pair would be one in which not only is the pair highly correlated with relevance, but the terms of the pair are more likely to co-occur together in the relevant documents than the non-relevant. The latter factor will emphasize strong independent contributions of the two terms to relevance. (If the latter factor is low, but the first factor is high, then the pair is probably a phrase and already included in the phrase component.)

Unlike in previous years, the choice of the final added single terms, phrases, or pairs, is deferred until after Rocchio weighting is done. At that time, the pairs with the highest Rocchio weights are kept (here, 100 single terms, 10 phrases, and 50 pairs). We also do not guarantee that original query terms will be kept in the final query. Our weighting schemes seem much more robust than in the past, so all this is feasible.

After the Rocchio (and possibly correlation) weighting is done and the query concepts have been selected, the weights are further optimized using a three-pass DFO algorithm. Using more passes as in TREC 4 does not seem to change performance, but is considerably more time consuming.

A summary of the steps used in routing, given a learning set, L, of judgements:

1. Create the initial vector query with *ltu* weighting from L.
2. Find the top 5000 documents to the query in L.
3. Expand query with single terms and phrases occurring in more than 5% or 10% of relevant documents
4. Weight expanded query using non-relevant documents from the query zone within the Rocchio formula.
5. Add pairs of co-occurring terms, one of which must be an original query term, occurring in more than 7% of relevant documents.

Run	Best	\geq median	$<$ median
Cor5R1cc	7	44	1
Cor5R2cr	5	42	3

Table 1: Comparative Routing Results (45 queries)

Run	Average Precision	Total Rel Retrieved	R Precision	Precision 100 docs
Cor5R1cc	.3842(.3807)	4249	.4137	.3296
Cor5R2cr	.3433(.3759)	4313	.3693	.3411

Table 2: Routing results - 45 (39) Queries

6. Weight pairs either according to Rocchio weights or correlation weights.
7. Restrict expanded query to 100 terms, 10 phrases, and 50 pairs, using those concepts with highest weights.
8. Perform a 3-pass DFO to fine-tune the weights.

There are obviously a large number of alternatives throughout this entire process. During the algorithm development we have only partially explored the options and expect we will alter the process in the future. This is especially true of the correlation weighting, which was added a few days before the routing deadline. There is much work to be done here!

Routing experiments and analysis

Cornell submitted two official routing runs, both automatic and both using the above process. Cor5R1cc uses co-occurrence of terms for pairs with the Rocchio weighting. As a more experimental run (i.e., we had no idea whether it would work) Cor5R2cr weights pairs with the correlation measure described above.

Table 1 shows that both runs performed very well and very consistently. Cor5R1cc was less than the median on only one query! Table 2 gives the absolute scores. The figures in parenthesis give the Average Precision for the 39 queries with more than 2 relevant documents. Cor5R2cr did poorly on two of those queries that Cor5R1cc did well on.

A head to head comparison shows the results were more different from each other than might be expected from the similarity of the averages. Each beat the other by more than 10% on exactly 10 queries.

If we break out some of the various components of the routing runs, we can get some insight as to what is important. The basic vector run in Table 3 starts at an average precision of .2462. Adding plain Rocchio reweighting of the original query terms improves results by 13%. Then there's a big jump of 19% when terms and phrases are added to go up to 100 terms and 10 phrases, reweighted using Rocchio. Adding co-occurrence pairs gains marginally, but then DFO on top of that is another 19% gain, with some large part of that being the reweighting of the co-occurrence pairs. (We don't yet know how to accurately weight those pairs.) All-in-all we end up with the routing judgements giving us a 56% improvement, which is much greater than in previous years. As can be seen from the last line in Table 3, our TREC 5 routing approach is a very substantial 23% improvement over the TREC 4 approach run on the same TREC 5 data.

One question that often comes up when talking about TREC routing is just how much of a problem is the mismatch between the learning set of documents and the test set of documents. Due to the difficulty of obtaining test collections, it is rare that the new TREC routing data can be considered a continuation of the old learning set data. We can't answer the question fully, but we ran the test set of documents as a separate full ad-hoc collection to get some ideas. The first two runs of Table 4 only differ in that the first run uses idf values determined from the learning set of documents, and the second run uses idf values determined from the test set of documents. There's a 6% difference between the two; certainly enough to have a noticeable effect, but probably not enough to pose serious problems to the testing methodology.

Run	Average Precision
Vector Lnu.ltu (including phrases)	.2452
Vector + reweighting	.2818 (+13%)
Vector + reweighting + exp	.3242 (+32%)
Vector + reweighting + exp + cooc:	.3365 (+37%)
Above, plus DFO (Cor5R1cc)	.3842 (+56%)
TREC 4 Routing on TREC 5 task	.3133

Table 3: Routing components results - 45 Queries

Run	Average Precision
Vector Lnu.ltu - idf of learning set	.2462
Vector Lnu.ltu - idf of test set	.2612
Ad-hoc expansion of test set	.2877

Table 4: Learning Set vs Test set Routing Runs - 45 Queries

We were also interested in just how much more the real learning set judgements help in expansion as opposed to the “fake” judgements used in our ad-hoc expansions that arbitrarily assume the top 20 to 30 retrieved documents are relevant. The third run of Table 4 gives the results of running our TREC 5 ad-hoc run (described below in the ad-hoc section) on the ad-hoc version of the test collection. As expected, the improvement over the basic vector run is much smaller. The .2877 figure can be compared against the .3365 result of the real relevance feedback without DFO (which obviously can’t be done in the ad-hoc case). The difference is 17%, certainly significant but somewhat disappointing considering the massive amount of relevance information available. This suggests that there is still room for improvement in our routing approaches; we should be able to do better.

Ad-Hoc Methodology

In TREC 4 we established a very good basic vector weighting model (“Lnu.ltu”) that we continue to use unchanged for TREC 5. Our TREC 5 efforts have been directed at improving our query expansion procedures. The overall expansion approach we use is to perform an initial retrieval upon the collection, retrieving a small number of documents. Those documents are assumed to be relevant and are submitted to a relevance feedback expansion and weighting stage similar to that described above in the routing section.

There are two steps in the above procedure on which we can improve. The first is to improve the initial retrieval so that the small number of documents assumed to be relevant are in fact more likely to be relevant. The second is to improve the expansion procedure once those relevant documents are obtained.

Mandar Mitra at Cornell has been working on the first step. The basic goal is to produce a “High-Precision” retrieved set: a set using retrieval techniques that may throw away lots of relevant documents in an effort to be sure that the remaining documents are likely to be relevant. This trading of recall for precision can be approached in many ways. The basic approach we use is “Query Coverage” (QC), parts of which has been worked on under various names by a number of groups like Xerox.

In “Query Coverage”, a query zone around the query is defined. These documents are candidates to be retrieved. Each document is submitted to an evaluation measure that attempts to determine how many query concepts are present within small windows of the document. If numerous unrelated query terms are discovered within a document window, then the multiple key concepts of the query are likely to be covered, and the document is likely to be relevant.

Note that many relevant documents may not satisfy the QC criteria. But since the goal here is precision, not recall, ignoring those documents will not hurt as long as other relevant documents are found.

The simplest sort of QC algorithm is to count the number of different matching terms between query

and document window. But this has the problem that phrases and strongly related terms can dominate. A three-word phrase match would be judged as important as if three distinct concepts matched. This happens often, so something more complicated is needed.

For TREC 5, our QC algorithm initially retrieves 1000 documents in the query zone. The correlations between all the original query terms are calculated within this zone. For example, for TREC query 248 (*What are some developments in electronic technology being applied to and resulting in advances for the blind?*), we find that the terms *advances* and *technology* are strongly related, whereas the terms *advances* and *blind* are not well related. Then a new query zone consisting of the top 50 documents initially retrieved is defined.

For each document in this tight query zone, the document is reindexed and broken up into 50 term, overlapping window, chunks. Each window is compared to the original query. The matching terms are considered in order of rarest first. Each matching term contributes its idf value times a factor inversely proportional to the maximum correlation the term has to all previous terms that have been considered in this window. I.e., if a term is highly correlated (among the top 1000 documents) with a previously considered term, then the match is not important and the idf value is deprecated. In the above example, if we have seen the term *advances* in a document, we would not consider the presence of the term *technology* to be a strong new match; but we will consider the presence of the term *blind* to be a strong new match, just because it is not well related to the previously seen terms.

The 20 documents with the highest matching windows are determined using the above algorithm. These documents are the High Precision result set, and will be assumed to be relevant in the next stage.

Note that this is in some sense the opposite of what the University of Massachusetts[8, 1] has done with their Local Content Analysis. They also break the document into chunks (but larger 500 word passages), but then reward terms highly correlated with the other query terms, whereas the approach tend to deprecate those terms. The approaches are actually not as contradictory as they seem; the UMass LCA process corresponds to our expansion using the entire document instead of our correlation step, which is a second-order correction step.

Analogously to our routing task approach, we want to define a non-relevant set of documents that are in the same domain as the query, though still non-relevant. The initial query zone of 1000 documents can be used for this; we make the assumption that documents ranked 500 through 1000 are non-relevant. For some queries with lots of relevant documents this may be a bad assumption, but it will often be reasonable.

The normal Rocchio expansion described in the routing section is then performed, except that the original query must be emphasized more, since the relevance information is much less dependable. Thus the original query weight counts as much as weights in the relevant and non-relevant documents: the *A.B.C* Rocchio parameters are set to values 8.8.8. Note that in previous years we used 8.8.0 and ignored the non-relevant documents altogether. We add 25 terms and 5 phrases to the original query, choosing those terms and phrases with highest Rocchio weight. Terms which occurred in fewer than 4 “relevant” documents and phrases which occurred in fewer than 2 “relevant” documents were ignored.

Our official automatic run Cor5A1se uses this as the final query. The other official automatic run, Cor5A2cr, continues adapting our full correlation routing approach to the ad-hoc task. Co-occurrence pairs are defined as in the routing task. They are weighted by a combination of the Rocchio weights of the terms, and the correlation of the terms within the top 1000 documents, with no dependence on relevance (again, the relevance information is much less dependable). The top 15 weighted co-occurrence pairs are added to query.

Ad-hoc automatic results and analysis

Our runs do comparatively well on the ad-hoc task, though not as well as they did last year. This year it is tougher to judge comparative performance from the median figures given all the fine-grained distinctions between runs. We complicated matters by “sneaking” an automatic run in under the manual category. As in past TRECs, we try to present one automatic run (Cor5A1se) that has been well developed over the past year, and one experimental run that gives a new untested approach that may or may not work (Cor5A2cr

Run	Task Pool	Best	\geq median	< median
Cor5A1se	Short Automatic	3	40	10
Cor5A2cr	Short Automatic	2	41	9
Cor5M1le	"Manual"	2	30	20
Cor5M1le	Long Automatic	7	42	8

Table 5: Comparative Automatic Ad-hoc Results (50 queries)

Run	Average Precision	Total Rel Retrieved	R Precision	Precision 100 docs
Cor5A1se (short)	.2065	2849	.2331	.2002
Cor5A2cr (short)	.2109	2848	.2404	.2018
Cor5M1le (long)	.2544	3092	.2845	.2272

Table 6: Ad-hoc results - 50 queries

as described above). These two short query runs took up our 2 automatic slots, so we put our long query automatic run (Cor5m1le) in a manual slot, where it did quite respectably against the real manual queries. Cor5m1le is exactly the same run as Cor5A1se, except it starts with the full version of the query instead of the Description field only.

Table 5 compares our three runs against the published medians of their respective task pools. Cor5M1le is compared against both the Manual pool, where it is officially located, and against the Long Automatic pool where it belongs. It would be best on 7 queries within the Long Automatic pool, and at or above the median on about 42 queries. (We say about 42 since the addition of Cor5M1le to the pool will change the medians by some unknown amount.)

These runs are all strongly above average, but they are not as consistent across queries as we would have hoped. This suggests our expansion techniques may have gotten off-target and expanded inappropriately. Further evidence of that comes from the low values of the absolute levels of retrieval effectiveness. If queries are harder, then the top 20 retrieved documents will contain fewer relevant documents. Thus it will be easier for the query expansion to sidetrack the final query into areas not related to relevance.

Table 6 shows the absolute levels of performance are very low; in fact lower than they have ever been in TREC. (TREC 1 official figures were worse, but only because they used a different evaluation methodology). The level of performance is due to the queries getting substantially harder rather than the systems deteriorating. A subjective look at the queries suggests that many queries this year are hard because they are much more high-level than normal. Later we present a table showing how each of the TREC 1-5 Cornell systems perform on each of the TREC 1-5 ad-hoc tasks. It clearly shows the queries getting harder every year, while the systems improve.

Another measure of the toughness of the various TREC ad-hoc tasks is seeing how well the best system for each query does. If no system does well on a query, then it can be called a hard query. You would expect these "best system" numbers to improve substantially each year as the systems improve, and as the number of participating systems increase. But as can be seen from Table 7, the numbers have remained constant for several years before taking a sudden dip this year. So systems have improved at about the rate that the queries have gotten harder until this year when the queries became much harder and outstripped the system improvements!

Table 8 looks at the components giving the performance on the ad-hoc task. The single term vector performance starts off at a low base-line value. Adding phrases helps only marginally, probably because the queries are all very short and not many phrases occur in them. Assuming the top 20 documents are relevant and reweighting the original query terms gives a substantial 9% improvement, as we can better tell what terms of the original query are extraneous and can be de-emphasized.

Another major improvement comes when terms from the top 20 documents are added to the query and weighted with Rocchio. This run corresponds to our TREC 4 expansion run, but with some minor tweaks. The TREC 4 result is given here for comparison sake, and indeed is almost identical.

TREC task	Average over all queries of Best run at 100 / MIN (NumRel, 100) (lower numbers imply poorer performance)
TREC 1	0.414
TREC 2	0.653
TREC 3	0.676
TREC 4	0.672
TREC 5	0.556

Table 7: TREC task hardness measure

Run	Average Precision
Vector, single terms only, Lnu.ltu	0.1484
Vector plus phrases	0.1506 (+1%)
Vector + reweighting	0.1619 (+9%)
Expansion, no non-rel docs, no reranking	0.1856 (+25%)
Expansion, 501-100 non-rel docs, no reranking	0.1909 (+29%)
Expansion, no non-rel docs, top50 reranked	0.2070 (+39%)
Expansion, 501-1000, top50 reranked (Cor5A1se)	0.2065 (+39%)
Expansion, 501-1000, top50, Corr Pairs (Cor5A2cr)	0.2109 (+42%)
TREC 4 Ad-hoc on TREC 5 task	.1833

Table 8: Ad-hoc components results - 50 Short Queries

The two major new attempts at improvement this year are using a query zone to define the non-relevant documents to use within the Rocchio formula, and reranking a close query zone of 50 documents to find the best 20 to use (basically a High-Precision filter). Using documents ranked 501-1000 gives a marginal improvement by itself, but this improvement disappears when the High-Precision top 20 documents are used. We do not know why this happens, but the improvement of using the 501-1000 non-relevant documents by itself was small enough so it may not be worth investigating.

The High-Precision reranked 20 document set gave a substantial improvement of 12% over the top 20 document set. We were very pleased with those results, especially since our investigations into High-Precision retrieval are just beginning. This offers great opportunities for improvement in the future.

Adding all the machinery to calculate co-occurrence pairs and weight with correlations did not yield any improvement at all. Cor5A2cr is better than Cor5A1se on 30 out of the 50 queries, but the differences were marginal. This needs to be explored further, but it is possible that better relevance information is needed.

Manual Ad-hoc relevance feedback run

For the first time, Cornell has submitted a run in the Manual Ad-hoc category. This is a repeat of the run we did last year in the Interactive track. Two expert users (authors of this paper) were given 25 queries each, and the initial retrieval rankings of the automatic run Cor5m1le, and told to judge relevance of documents for 5 minutes per query. On average, each user looked at about 25 documents per query, though a large number of those documents were easily dismissed. These relevance judgements were then used within the Rocchio framework to construct a new feedback query, expanding by 25 terms and 5 phrases, but with no co-occurring pairs. This new query was then automatically run from scratch (no frozen ranks of the user judged documents), and the top 1000 documents submitted to NIST. No other manual effort was made, and the user judgements were used only to construct the new query.

The only manual input was relevance judgements, which any user should be able to make accurately. Thus the expertise of the users should not be a factor here, other than a complete system novice might take longer because of unfamiliarity with the user interface.

Run	Average Precision	Total Rel Retrieved	R Precision	Precision 100 docs
Cor5M1le (automatic)	.2544	3092	.2845	.2272
Cor5M2rf (manual)	.2931	3085	.3112	.2412

Table 9: Comparison of manual vs. automatic Ad-hoc 50 queries

The results in Table 9 show we got a 15% improvement overall. The relevance feedback improved performance on 37 out of the 50 queries. There were two queries in which the users could find no relevant documents.

The 15% improvement is very substantial, showing the effectiveness increase that any user should be able to achieve with relevance feedback. In practice, the improvement should be even more since these test result values are lowered due to the well-known mismatch between the test user's notion of relevance in the on-line judgements, and the assessor's notion of relevance for evaluation. We have not yet studied the mismatches between the two, but plan to in the future.

Spanish

The multi-lingual track demonstrates the language-independent nature of SMART. The techniques used in this year's ad-hoc task were directly applied to the Spanish task. The modifications needed to enable retrieval in Spanish were done for TREC 3 [5] and took a total human time of 5 – 6 hours.

Cor5SP1s and Cor5SP2l

Cor5SP1s and Cor5SP2l, the two runs submitted by Cornell, use identical techniques. The only difference is that Cor5SP1s indexes only the Spanish description field (<S-desc>) of the query, while Cor5SP2l indexes the Spanish narrative field (<S-narr>) of the query as well. The queries were weighted with *ltu* weights, and the documents with *lnu* weights. 1000 documents were retrieved using simple inner product similarity with the initial query. The 50 top-ranked documents were re-ranked as in the ad-hoc run Cor5A1se. The top 20 documents in the resulting ranking were assumed to be relevant and documents ranked 501 – 1000 were assumed to be non-relevant. Rocchio relevance feedback was then used to expand and reweight the query (25 single terms and 5 phrases were added). The 1000 documents retrieved using this query constitute our official submission.

Spanish Results

The results for the two runs are shown in Table 10 and Table 11. Both runs did well compared to the median score of the groups participating in the Spanish task. As expected, the absolute figures are better when long queries are used.

Run	Best	\geq median	< median
Cor5SP1s	1	18	7
Cor5SP2l	3	18	7

Table 10: Comparative Spanish Results - 25 Queries

Table 12 shows the average precision obtained at each stage of the Cor5SP1s run. A simple vector match using single terms only yields an average precision of 0.3154. Our official result (0.3949) is 25.2% above this baseline. The improvement is mostly due to the addition of new terms to the query from the top-ranked documents. When the query is not expanded, but simply re-weighted using Rocchio feedback under the assumption that the 20 top-ranked documents are relevant, the results do not improve appreciably¹.

¹ The improvement due to re-weighting is greater (5.3%) for the long queries (Cor5SP2l).

Run	Average Precision	Total Rel Retrieved	R Precision	Precision 100 documents
Cor5SP1s	.3949	2103	.3890	.3448
Cor5SP2l	.4323	2222	.4468	.3796

Table 11: Spanish Ad-hoc - 25 queries

For Cor5A1se, the English equivalent of Cor5SP1s, the improvements obtained are compatible — the average precision for the submitted run (0.2065) is 39.2% above a lower baseline of 0.1484 corresponding to the base vector run.

Base vector single terms only	Base vector single terms and phrases	Rocchio re-weighting no expansion	Rocchio expansion
3154	3091 (-2.0%)	3170 (+0.5%)	3949 (+25.2%)

Table 12: Analysis of Cor5SP1s

Spanish Discussion

Phrases are constructed purely statistically — any pair of adjacent non-stopwords that occur sufficiently frequently in the corpus are deemed to constitute a phrase. Thus, the method used to generate phrases for our English collection was used to generate phrases for the Spanish collection as well. The use of phrases did not improve retrieval, however. For the base vector run using inner product similarity with the initial queries, the average precision dropped from 0.3154 to 0.3091 when phrases were used. Similarly, expansion by 25 single terms and 5 phrases (our official submission, which gave an average precision of 0.3949) is no better than expansion by 25 single terms only (average precision 0.3951).

A similar observation holds for this year's English runs as well. The use of phrases in the simple vector match run for Cor5A1se yields an improvement of only 1.5% compared to about 5% in previous years, and the use of phrases during automatic expansion improves performance by only 1.4% as against about 12% in previous years. We need to investigate why the usefulness of phrases seems to have diminished.

Accented characters occur inconsistently in the AFP corpus i.e. the same word occurs both with and without an accent on some letter. We need to investigate ways to handle these inconsistencies.

Chinese

If the linguistic effort Cornell spent on Spanish was small, the effort spent on Chinese was minuscule. There were no people involved at Cornell who understand anything about the Chinese language. We knew that Chinese characters occupied two bytes, and we discovered that punctuation all began with one of two byte values. Other than that, we can do no linguistic analysis and therefore can do no segmentation or stopwords. Every Chinese character (except punctuation) is retained and treated as a separate word within SMART.

The changes to SMART for chinese involved about 30 lines of code in the tokenizer, and two lines of code in the phraser (phrase components are not separated by whitespace in Chinese). Other than that, no changes were made to SMART or our basic retrieval algorithms. The most human time spent on the Chinese track was spent finding a Chinese text pager on the Web, and getting it to work.

The Chinese collection is indexed in the same fashion as English. A list of Chinese “phrases” are automatically formed by taking all adjacent pairs of non-punctuation characters that occur more than 20 times in the test collection. This list is deliberately made larger than our English list of phrases (174,891 entries as opposed to 158,099 entries) since we know that no Chinese word segmentation will be used.

Other than the larger list of phrases, we index Chinese exactly like English, and our retrieval runs corresponded exactly to the runs we would do in English, including weighting and ad-hoc expansion. The Cor5C2ex run repeats the Cor5A1se exactly (including determining the top 20 documents by reranking the

Run	Best	\geq median	$<$ median
Cor5C1vt	1	24	4
Cor5C2ex	5	28	0

Table 13: Comparative Chinese Results (28 queries)

Run	Average Precision	Total Rel Retrieved	R Precision	Precision 100 docs
Cor5C1vt (vector)	.3266	1286	.3598	.3026
Cor5C2ex (expansion)	.3598	1343	.3829	.3084

Table 14: Chinese Ad-hoc - 19 queries

top 50 with a High-Precision filter) except we expand by 15 single terms and 15 phrases instead of 25 single terms and 5 phrases. The other official run, Cor5C1vt, is a straight simple Lnu.ltu weighted vector run.

Our non-linguistic approach does remarkably well. Table 13 shows that the Cor5C2ex run is at or above the median for all 28 queries, having the best results on 5 queries. Even the simple vector match is only below the median on 4 queries.

Tables 14 and 15 give more details of the evaluations of the runs and the components going into the runs. The expanded query does 10% better than the basic vector version. That is much less than the improvement on the English short query run. However, starting with a long English query, expansion only improves the English result by 14% (.2234 for a base vector with phrases as compared to .2544 for Cor5M1le). We only ran the long versions of the Chinese queries, so the difference in percentage improvement between Chinese and English is not all that great.

One interesting result that bears further investigation is that expansionless reweighting of the original query terms and phrases based on the top retrieved documents actually hurts performance. The third run of Table 15 is 5% worse than the second run. This compares to a 5-10% typical improvement in English and Spanish. One conjecture is that very common single characters start dominating the phrases, but this remains un-examined.

Comparison with past TREC's

It is difficult to determine how much systems are improving from TREC to TREC since the queries and the documents are changing. For example, in TREC 3 the "Concept" field of the queries was removed. These terms proved to be very good terms for retrieval effectiveness in TREC 1 and TREC 2; thus the TREC 3 task without them is a harder task than previous TRECs. The TREC 4 task was more difficult since so much more of the text was removed from the queries. As was discussed earlier in this paper (see Table 7) the TREC 5 task is even more harder than TREC 4, apparently because of the types of queries.

To examine both how much SMART has improved over the years of TREC, and how much harder the TREC ad-hoc tasks have gotten, we ran our 5 TREC SMART systems against each of the 5 TREC ad-hoc tasks. Table 16 gives the results. Note that the indexing of the collections has changed slightly over the years so results may not be exactly what got reported in previous years. In the interest of speed, we ran our current implementation of the query and document indexing and weighting. Things have changed more

Run	Average Precision
Vector, single terms only, Lnu.ltu	0.2859
Vector plus phrases (Cor5C1vt)	0.3266 (+14%)
Vector + phrases + reweighting	0.3063 (+7%)
Expansion, 501-1000, top50 reranked (Cor5c2ex)	0.3598 (+26%)

Table 15: Chinese components results - 19 Queries

Methodology and Run	TREC 1 Task	TREC 2 Task	TREC 3 Task	TREC 4 Task	TREC 5 Task	% Change from TREC 1 task
TREC 1: ntc.ntc	.2431	.2594	.2095	.1507	.1038	-57
TREC 2: Inc.ltc	.3078	.3352	.2801	.1626	.1072	-65
TREC 3: Inc.ltc-Exp	.3474	.3534	.3209	.2022	.1254	-63
TREC 4: Lnu.ltu-Exp	.3609	.3762	.3760	.2841	.1833	-49
TREC 5: Exp-rerank	.3765	.3830	.3972	.3146	.2065	-45
% Change from ntc.ntc	+55	+48	+90	+109	+98	

Table 16: Comparisons of past SMART approaches with present

than we expected; a number of the figures from the earlier runs on the earlier collections are off by as much as 3–5%, probably due to the indexing changes. The results, though, are all consistent with each other.

The last column of Table 16 gives an indication of how much harder the TREC task has gotten during the 5 years of TREC. Five quite different versions of the same system all do from 45% to 65% worse, in absolute numbers, on the TREC 5 task as compared to the TREC 1 task. The TREC 1 and TREC 2 figures are about the same. Performance starts to drop in TREC 3 and 4 when the queries got progressively shorter. The short high-level queries of TREC 5 proved very difficult for all versions of SMART.

A couple of points of interest about our results in Table 16. Our TREC 5 approach does not get as much improvement on the TREC 1 and TREC 2 tasks as it does on later tasks. The TREC 5 expansion, instead of adding a specified number of terms and phrases as in previous years, specifies what the final length of the vector should be. That leaves little room for new term expansion on the very long queries of TREC 1 and TREC 2. The results here suggest we need to go back to our earlier expansion strategy for long queries.

The Inc.ltc weighting strategy becomes less effective with the short queries of TREC 4 and TREC 5 (as compared to ntc.ntc). One reason is simply the tf component of the query weight is useless with the short queries. But in addition to that, there are indications that idf is more important in short queries than long. This needs to be studied in the future.

I am still somewhat surprised that we and other groups can continue to improve our systems as consistently as we have been able to. It really is impressive!

Conclusion

The Cornell SMART Project is again a very active participant in this year's TREC program. With the exception of our one relevance feedback manual run, everything we have presented here is completely automatic and uses no outside knowledge base (other than a small list of stopwords to ignore while indexing). Manual aids to the user can be built on top of this system to provide even greater effectiveness.

Most of our effort this year was spent on the routing task. Overall, our routing performance improves by 23% over last year's approach. We re-examined our parameters and introduce the idea of "query zones" which contain documents which may be related to the domain of the query, but may not be relevant. Terms useful in distinguishing the query zone documents from the rest of the collection may not be useful in distinguishing relevant from non-relevant documents within the collection.

We also add dynamic pairs to our expansions. These are a high precision device intended to separate the relevant documents from the top non-relevant documents. Pairs of terms that co-occur in the relevant documents are chosen to be added to the query.

Once again, DFO (Dynamic Feedback Optimization) substantially improves routing retrieval performance. The combination of the added dynamic pairs and DFO worked very well (suggesting that possibly our initial weights on the dynamic pairs could be substantially improved in the future).

Our ad-hoc expansion approach improves 12% over last year's very impressive results. The major source of the improvement is reranking the top 50 initially retrieved documents using a High-Precision approach to end up with 20 documents that individually cover the concepts of the query. These documents prove to be

more useful as a source for expansion and reweighting information.

Our manual results show that minimal involvement by users, just having users judge the relevance of documents, can result in a very effective retrieval set. Any user can judge whether documents are useful; that is their purpose in coming to the information retrieval system in the first place.

The Spanish results are again considerably above the median, and used no language knowledge. The runs were exactly the same runs as we run on the English tasks.

The Chinese results are also extremely good and are done with no knowledge at all of Chinese or of Chinese word segmentation. This may suggest that segmentation is a minor issue for retrieving Chinese and shouldn't be a major focus.

We had been improving our ad-hoc results by 20-25% a year. We concentrated on routing and fell a bit behind that figure for ad-hoc this year. Maybe next year...!

References

- [1] James Allan, Jamie Callan, W. Bruce Croft, Lisa Ballesteros, John Broglio, Jinxi Xu, and Hongmin Shu. INQUERY at TREC-5. In D. K. Harman, editor, *Proceedings of the Fifth Text REtrieval Conference (TREC-5)*. NIST (in preparation), 1997.
- [2] Chris Buckley, James Allan, and Gerard Salton. Automatic routing and ad-hoc retrieval using SMART : TREC 2. In D. K. Harman, editor, *Proceedings of the Second Text REtrieval Conference (TREC-2)*, pages 45-56. NIST Special Publication 500-215, March 1994.
- [3] Chris Buckley and Gerard Salton. Optimization of relevance feedback weights. In Ed Fox, Peter Ingwersen, and Raya Fidel, editors, *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 351-357, New York, July 1995. ACM.
- [4] Chris Buckley, Gerard Salton, and James Allan. Automatic retrieval with locality information using SMART. In D. K. Harman, editor, *Proceedings of the First Text REtrieval Conference (TREC-1)*, pages 59-72. NIST Special Publication 500-207, March 1993.
- [5] Chris Buckley, Gerard Salton, James Allan, and Amit Singhal. Automatic query expansion using SMART : TREC 3. In D. K. Harman, editor, *Overview of the Third Text REtrieval Conference (TREC-3)*. NIST Special Publication 500-225, 1995.
- [6] Chris Buckley, Amit Singhal, and Mandar Mitra. New retrieval approaches using SMART : TREC 4. In D. K. Harman, editor, *Proceedings of the Fourth Text REtrieval Conference (TREC-4)*. NIST Special Publication 500-236, 1996.
- [7] Hinrich Schutze, David A. Hull, and Jan O. Pedersen. A comparison of classifiers and document representations for the routing problem. In *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 229-237, 1995.
- [8] Jinxi Xu and W. Bruce Croft. Query expansion using local and global document analysis. In *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 4-11, 1996.

INQUERY at TREC-5

James Allan, Jamie Callan, Bruce Croft
Lisa Ballesteros, John Broglio, Jinxi Xu, Hongmin Shu

Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, Massachusetts 01003, USA

The University of Massachusetts participated in five tracks in TREC-5: Ad-hoc, Routing, Filtering, Chinese, and Spanish. Our results are generally positive, continuing to indicate that the techniques we have applied perform well in a variety of settings.

Significant changes in our approaches include emphasis on identifying key concepts/terms in the query topics, expansion of the query using a variant of automatic feedback called “Local Context Analysis”, and application of these techniques to a non-European language.

The results show the broad applicability of Local Context Analysis, demonstrate successful identification and use of key concepts, raise interesting questions about how key concepts affect precision, support the belief that many IR techniques can be applied across languages, present an intriguing lack of tradeoff between recall and precision when filtering, and confirm once again several known results about query formulation and combination.

Regrettably, three of our official submissions were marred by errors in the processing (an undetected syntax error in some queries, and an incomplete data set in another case). The following discussion analyzes corrected runs as well as those (not particularly meaningful) submitted runs.

Our experiments were conducted with version 3.1 of the INQUERY information retrieval system. INQUERY is based on the Bayesian inference network retrieval model. It is described elsewhere [5, 4, 12, 11], so this paper focuses on relevant differences to the previously published algorithms.

1 Ad-Hoc Experiments

The emphasis in this year’s Ad-Hoc experiments was on how to create very effective queries from short natural language topics. In particular, how to create *several* rather different queries from the Description field of the topic. The underlying assumption, based on past experience, was that a combination of several “pretty good” queries would be more effective (or, at least easier to create) than one “excellent” query.

Three techniques were explored. The first, which we call *basic query processing* in this paper, is an extension of the query processing techniques we used in previous TREC evaluations. The second, which we call *core query processing*, attempted to identify and emphasize the most critical query terms. The third, *local context analysis*, is a new approach to query expansion [14]. Each technique is discussed in more detail in the subsections that follow.

Two Ad-Hoc runs were submitted. Each query in INQ301 was a combination of three queries created from the Description (<desc>) fields of topics 251-300. Each query in INQ302 was a

combination of two queries created from the Description, Narrative (<narr>), and <title> fields of the topics.

Experiments were run on TREC volumes 2 and 4, using the released version of Inquiry 3.1. No special changes were made for TREC. The most significant changes in Inquiry relative to last year's TREC experiments is the adoption of the Okapi term frequency formula ($\frac{tf}{tf+K}$) [10] and of the Kstem dictionary-based stemming algorithm.[8]

1.1 Basic Query Processing

As in past years, our *basic query processing* is a series of processing steps that produce a "words" part of the query and a "phrase" part of the query.

The "words only" portion of the query is generated by removing negation and "stop" phrases (e.g., "To be relevant, a document should contain"), converting hyphenated terms into an OR of hyphenated and un-hyphenated forms, and recognition and expansion of country names.

The "phrase" portion of the query is initially processed in a similar manner, except that words are not discarded immediately, because doing so would interfere with later syntactic processing. Instead, they are marked as "stopped" for later deletion.

Syntactic processing is based on a few assumptions about full-text queries: when a purpose clause is present, it is the most important part of the query. After that, there are certain prepositions that tend to signal significant concepts, for example, "of" and "for."

In addition, different kinds of questions require slightly different treatment. In queries of the form "What is the X of Y?" and "How X is Y?", Y tends to be the important concept. For the incomplete sentence "X done by Y for Z", Z and Y are likely to be significant. However, a laconic sentence fragment such as "X of Y" suggests equal treatment for both X and Y. Weighting of terms reflected all of this. In addition, phrases were built out of significant noun groups. Time constructions were for the most part deleted or down-weighted. Unless a date is spelled out and found in the focus position (e.g., "What happened on month day, year?") it is unlikely to be reliable.

In order to do this processing, the queries were tagged with part-of-speech information and a loose parse was constructed, based on constituents such as noun phrases and prepositional phrases. Conjunctions were handled as completely as possible so that a preposition with multiple objects would receive credit for each object, as in "the cost of military outfitting and equipping." Some kinds of appositives were recognized, such as acronyms.

Two shortcomings of the current state of development:

1. No advantage is taken of groupings revealed by constituents and conjunctions beyond that of noun group phrasing; and
2. Enumerative appositives are not distinguished and handled.

The latter problem is not difficult to address. The major issue is to distinguish whether appositives are restrictive, adding little new information, or whether they give additional examples of subjects of interest. The surprising thing is that a great deal can be done with syntax alone, time permitting. A lexicon with even simple argument structure, or use of collection statistics to infer the same information, would enhance the syntactic analysis at crucial decision points.

1.2 Core Query Processing

The *core query processing* was an extension of basic query processing intended to improve precision at low recall. It is based on the observation that in many queries, there are one or two terms

that *must* be in a document for the document to be relevant. Our goal was to identify such terms automatically, and then to ensure their presence in the top-ranked documents.

The procedure used to select the required, or *core*, terms was elaborate, and is probably more complex than necessary. The goal was to apply several forms of (uncertain) evidence to identify core terms. The procedure consisted of the following steps applied to the Description field of the topic.

1. Discard words on the query stopword list (e.g., “relevant”, “document”). Rank the remaining terms by $w \times avg_tf^{0.7} \times idf$, where w is the basic query processing weight, avg_tf is the term’s average frequency in documents where it occurs, and idf is Inquery’s inverse document frequency.¹ The top ranked term is considered the *core* term.
2. If the core term is part of a phrase identified by basic query processing, the phrase becomes the core term.
3. Cluster the query terms using EMIM, as in [6]. If the cluster contains multiple terms, replace the core term with a proximity operator (unordered window) containing the clustered terms. The size of the window ranged from 20 to 200, depending upon the number of matches in the corpus (smaller when there are many matches).
4. If the term weighted most highly by basic query processing is not in the core cluster, add it.
5. If the proximity operator matches fewer than 10 documents, it is too strict. Relax it by discarding terms with low weights until at least 10 documents are matched.

Although complex, this procedure consists of essentially three steps. Query terms are ranked by two methods (one statistical, one linguistic), and clustered by one. The results are combined to produce a proximity operator of core terms that retrieves at least 10 documents, but preferably not too many more. If this approach proves useful, it can be made substantially simpler and more efficient.

The set of documents retrieved by the proximity operator is always a subset of the documents retrieved by basic query processing. Documents are ranked within each set by their basic query processing scores. The two sets are then combined such that documents retrieved by the proximity operator (a small set, we hope) are ranked ahead of documents retrieved by basic query processing.

1.3 Local Context Analysis

Local Context Analysis (LCA) is an “automatic feedback” technique that expands the query by including terms which occur in the top-ranked documents, but only if they occur near query terms. It is explained in detail elsewhere.[14]

LCA first runs a query against the database to get the top n documents. The concepts from those documents are ranked based on the their co-occurrence (in the top ranked documents) with the individual query terms. (The hypothesis is that good expansion concepts tend to co-occur with all query words in the top ranked documents.) The top ranked m concepts are used for query expansion.

For the Ad-hoc track, we used the 100 top-ranked 300-word passages rather than the entire document. Concepts that were available for expansion were noun phrases as recognized by Jtag[13]. 70 concepts were added to each query with tapering weights w_i for concept i :

$$w_i = 1.0 - 0.9 * (i - 1)/70$$

¹The approach was inspired by Kwok[7], but the formula is original.

1.4 Combining queries

The three types of queries—basic, core, and expansion—were combined in two different ways. For INQ301, the automatic run starting from short (description only) topics, the combination was:

```
Expanded Query = #wsum( 1.0 1.0 basic-query  
                        2.0 core-query  
                        4.0 expansion-query)
```

That is, the belief from the expanded portion of the query was four times as strong as that from the basic query.

For INQ302, the basic query processing used the description and narrative sections, and the core concept processing was omitted. Those queries were formed as:

```
Expanded Query = #wsum( 1.0 1.0 basic-query  
                        2.0 expansion-query)
```

2 Routing Experiments

We submitted only one run in the routing track, and made few changes from last year's run; our focus was on work in the other tracks. The differences between this year and last were in original term weighting, application of Dynamic Feedback Optimization, and INQUERY's belief score.

Queries were expanded by adding terms, adjacent word pairs, and nearby word pairs. The selected concepts were chosen from a large candidate set by comparing their occurrences in relevant and non-relevant training documents. Weights were assigned using a Rocchio combination of "term belief". Finally, the weights were adjusted by fitting them more closely to the training data using a technique very similar to one described by Buckley and Salton[2].

For the final run, as required by the routing guidelines, collection wide statistics such as "idf" and "average document length", were taken from the training collections. Collection frequency information in the test database was not used in any way.

2.1 Term selection

Training data for routing came from TREC disks 1–3 as well as the TREC-4 routing disk. For each query, we used all known relevant training documents for that query as well as the same number of top-ranked non-relevant documents identified by running the original query against the training databases which had previously been judged for that query. Because we consider any unjudged document to be non-relevant, this process required building 5 training databases. (The "original query" refers to the result of automatically creating an INQUERY structured query from the original TREC topic, without reference to relevance judgments. A more complex version of that process is discussed in the Ad-hoc description above.)

For each query, all terms occurring in at least 5 relevant documents were identified, and were then ranked by their relative occurrences in the relevant and non-relevant documents. That is, by:

$$\frac{df_{rel}}{n_r} - \frac{df_{nonrel}}{n_{nr}}$$

where df_{rel} is the total number of relevant documents containing the term, df_{nonrel} is that count in non-relevant documents, n_r is the number of relevant documents, and n_{nr} is the number of

non-relevant (recall that here $n_r = n_{nr}$). The top 50 terms in that order were chosen and weighted using a Rocchio formula:

$$\beta \cdot \frac{1}{df_{rel}} \sum_{rel} belief - \gamma \cdot \frac{1}{df_{nonrel}} \sum_{nonrel} belief$$

where $\beta = 8$, $\gamma = 2$, and the belief for term t in document d was calculated by the formula:

$$\frac{tf_{t,d}}{tf_{t,d} + 0.5 + 1.5 \frac{doclen}{avgdoclen}}$$

where $tf_{t,d}$ is the number of occurrences of term t in document d , $doclen$ is the length of document d in words, and $avgdoclen$ is the average length (in words) of the documents in the training collection for this query. This equation is the “tf” portion of the belief function used by INQUERY version 3.1; it was adopted from Okapi[10].

The use of top-ranked non-relevant documents for negative feedback information is the approach that we used last year also,[1] preferring to balance the number of relevant and non-relevant used in training. An alternate approach is to use all of the collection other than the known relevant documents for negative information. Buckley et al[3] have recently adopted an approach similar to ours which they call “query zoning.”

2.2 Additional concepts

The same process described above was applied to find concepts based upon pairs of terms also. This is the same technique that we applied last year.[1] In this case, candidate pairs were found considering *only* the 200-word passage of the training document which best matched the original query. From those passages, 50 adjacent term pairs (ordering significant) were chosen. In addition, 50 each of word pairs within 5, 20, or 50 words (order insignificant) were added. Selection and weighting were done exactly as described above.

In all, each query was augmented with 250 new concepts, though there was some overlap. In query 240, for example, “anti terror” appeared in every category.

To create the expanded query, the original query was first flattened to remove some of its structure. Phrases, pairs of words in proximity, synonym operators, and other “concept creation” operators were left intact, but evidence-combining operators such as #sum were removed. The query was then a set of concepts which were each assigned a weight of 10 times the number of occurrences of that concept in the original query.

The 250 new concepts were then added to that list and assigned the weight as described above. Note that this means that original query terms tended to have a weight that was 2-3 times that of the new concepts. “Mistakes” in that weighting were corrected in the next step.

Buckley et al[3] explored term co-occurrences in their recent work, too. They used twice as many non-relevant documents for statistics-gathering, and used different measures for term selections, but the idea is similar.

2.3 Weight adjustments

We again used the Dynamic Feedback Optimization approach of Buckley and Salton[2] to adjust the chosen weights to achieve higher effectiveness in the training data, predicting that this effort will result in better effectiveness in the test documents.

The approach starts by evaluating the query on the training data. Then some concept's weight is adjusted and the slightly different query is evaluated. If the effectiveness has improved, the change is retained; if the new weight hurts effectiveness, the original is restored. In both cases, the next concept weight is tried. This process repeats until all concept weights have been "tweaked" once, completing a pass.

In TREC-4 we repeated each pass until no more improvement was found, but that caused a small problem with overfitting. For TREC-5, we reverted to Buckley and Salton's original approach and just ran a pass of adjustments once for each possibility: increase the weight by 100%, then 50%, 25%, 12.5%, and finally 6.25%. So each concept had the opportunity to be adjusted 5 times.

For efficiency reasons, the evaluations were done using only the top 20,000 documents retrieved from that query's training set in response to the new query (prior to reweighting). Effectiveness of a "tweak" was measured by average precision among those 20,000 documents.

3 Spanish Experiments

The Spanish retrieval experiments built upon the work done in TREC-4 [1]. This year's focus was on combining global analysis and local feedback for query expansion via Local Context Analysis as described in Section 1.3.

3.1 Query Formation

Query processing for the Spanish topics is similar to that of the English topics and was used to generate base queries for retrieval. Base queries were then expanded using Local Context Analysis. When expanding, the top 31 concepts were added with multi-term concepts wrapped in a **#phrase** operator with the restriction that all terms be found within a window of 25 terms. The top concept was given a weight of 1.0 with all subsequent concepts down-weighted by 1/100 for each position further down in the rank (so the second one had a weight of 0.99, then 0.98, and so on).

Concepts (single terms or phrases) containing terms occurring in the original query (called "duplicate phrases") and phrases recognized in the original parsing of the base queries (called "query phrases") are considered to be more important. These concepts were identified and given more weight.

Two sets of queries were generated, one using only topic descriptions (SIN300) as the base for expansion and the other using both descriptions and narratives (SIN301). The original query and additional concepts were combined in two ways with the following basic structure:

```
#wsum(1.0 1.0 original-query
      4.0 LCA-concepts
      4.0 duplicate-phrases and query phrases)
```

For run SIN300 the third group contained both duplicate phrases and query phrases. The SIN301 run was done similarly, but the third group of concepts contained only duplicate phrases.

4 Chinese Experiments

INQUERY required few modifications to support Chinese information retrieval.

1. Each Chinese character is indexed as a term. Exceptions are made for the characters making up numbers and the elements of dates, which are indexed as a group.

2. Queries are segmented to separate them into words. The hidden Markov model segmenter[9] is used for this purpose.

4.1 Segmentation

To allow for flexibility in segmentation at query time, the Chinese documents were indexed by treating each Chinese character as an individual term.

The queries were then segmented automatically with a segmenter based upon hidden Markov models[9]. In the resulting query, each segmented Chinese word was represented by a proximity operator which required the glyphs in the word to be located immediately adjacent and in order.

To compensate for possible segmenter errors, sequences of single characters were grouped together in a *#phrase* operator, which is evaluated as a proximity of #3 if the group of glyphs appears commonly in the textbase, or as a sequence of single glyphs if the sequence is infrequent. Single characters not found in a sequence were downweighted.

4.2 Query Construction

Title and Description fields were used. Each word in the description was weighted as a single term (except for isolated single characters), but the whole title was weighted as one term. The segmenter was augmented with a name recognizer to reduce errors of name segmentation.

The queries were then expanded by a slight modification of the Local Context Analysis method described in the section on Adhoc English query processing[14]. For purposes of expansion, any segmented “word” in the top-ranked documents was a candidate. Query concepts were segmented words found in the title or description text of the topic. Concepts for expansion were taken from the 10 top-ranked documents and only 30 concepts were added. Concept i was assigned weight:

$$w_i = 1.0 - (i - 1)/60$$

Two runs were done to test the effect of the expansion terms. One run (called HIN300) weighted the expansion section at half the weight of the original query. The other run (HIN301) gave the expansion section the same weight as the original query. (In the English Ad-hoc track, the expansion terms were given two or four times the weight of the original query.)

5 Filtering Experiments

Our approach to the filtering track leveraged heavily off of the routing track’s results. After the final routing queries were generated (as described above), we ran each query on its *training* database. For a given query, we then found the point in its ranked list which yielded the optimum value for each of the styles of filtering—that is, where including all documents ranked from 1 to that point resulted in a maximum utility score.

We then ran the queries on the *test* database and for each style of filtering, used the calculated threshold to decide which documents were filtered. Because any collection-specific information that was used in ranking (e.g., idf and average document length) was taken from the training database, the scores were directly comparable. Clearly, if the test database’s idf value were used, this approach could not possibly work.

Note that the submitted documents were therefore top portions of the same ranked lists submitted for the routing run—with the exception that in some cases more than 1,000 documents were filtered because the threshold was so low.

	nl	B*	C*	B+C	L*	B+L	C+L	B+C+L*
1-1-1	0.1442	0.1796	0.1852	0.1851	0.1604	0.1919	0.1990	0.1964
1-2-4	—	—	—	0.1851	—	0.1801	0.1978	0.2001
1-2-4?	—	—	—	—	—	—	—	0.2046

Table 1: Average precision breakdown for various combinations of runs. nl is the description only, B is basic query processing, C is core concepts, and L is the LCA. Table 2 shows more detail on the starred runs. BCL is the same as INQ301. Rows show different weights assigned to B, L, and C components; “4?” means either 4 or 0, whichever is best.

6 Ad-Hoc Results and Discussion

Two sets of results, INQ301 and INQ302, were evaluated in the ad-hoc document retrieval evaluation. Both sets of results were based on completely automatic processing of the TREC topic statement into two queries, and automatic query expansion. The difference is that INQ301 was formed from the Description (<desc>) field only, while INQ302 was formed from both the Description and Narrative (<narr>) fields.

A query processing error caused every query in INQ302 to reduce to the useless, but valid, query “2.0”. Hence the output from that run is also useless: all it demonstrates is that UMass did not examine any documents prior to submitting results to NIST. In the table below, INQ302c is an unofficial run showing what we intended to submit (the “corrected” run).

The INQ301 adhoc run (“Description only”) was treated as the baseline run. The official evaluations, and one unofficial run, are summarized below, using the results for all 50 queries.

Query Type	Average Precision			
	5 Docs	30 Docs	100 Docs	11-Pt Avg
INQ301	.38	.28	.17	.20
INQ302	.02 (−95.7%)	.01 (−98.3%)	.00 (−98.5%)	.00 (−97.9%)
INQ302c	.40 (+7.5%)	.30 (+9.6%)	.21 (+21.0%)	.23 (+13.3%)

As expected, using the Narrative field to create queries is helpful, producing a 13.3% improvement in average precision. We do not know yet whether the improvement is due to creating an improved base query, or whether it is due to improved query expansion.

Tables 1 and 2 show the contributions of the various components in the INQ301 queries, as compared with a baseline using all (and only) the words in the Description field. Both basic and core query processing techniques produced significant improvements in precision at all levels of recall. Query expansion terms/phrases, when used by themselves as a query, are also significantly better than the words in the Description field at all recall levels, although they are significantly worse at ranks 1–20.

It is somewhat surprising that the core query processing technique improved recall instead of precision (as compared with basic query processing). This technique was intended to “promote” a small number of documents that contain key query terms. The danger was that it would promote many irrelevant documents. The results suggest that it promoted a few extra irrelevant documents, depressing low-recall precision slightly. It is not immediately clear how it improved precision at high recall.

As expected, a weighted combination of basic query processing, core query processing, and local context analysis terms/phrases was more effective than any of its components at 10% recall and

Recall	Precision (50 queries)								
	Desc Only	Base Query Processing (B)			Core Query Processing (C)		Local Context Analysis (L)		INQ301 (B+C+L)
0	0.5644	0.6482	(+14.8)	0.6449	(+14.3)	0.4779	(-15.3)	0.5919	(+ 4.9)
10	0.3294	0.3733	(+13.3)	0.3808	(+15.6)	0.3097	(-6.0)	0.3811	(+15.7)
20	0.2593	0.2921	(+12.6)	0.2817	(+ 8.6)	0.2670	(+3.0)	0.3071	(+18.4)
30	0.2071	0.2399	(+15.8)	0.2419	(+16.8)	0.2259	(+9.1)	0.2586	(+24.9)
40	0.1650	0.2055	(+24.5)	0.2229	(+35.1)	0.1987	(+20.4)	0.2359	(+43.0)
50	0.1220	0.1739	(+42.5)	0.1746	(+43.1)	0.1727	(+41.6)	0.1955	(+60.2)
60	0.0781	0.1193	(+52.8)	0.1253	(+60.4)	0.1322	(+69.3)	0.1682	(+115.4)
70	0.0523	0.0861	(+64.6)	0.0913	(+74.6)	0.0961	(+83.7)	0.1138	(+117.6)
80	0.0350	0.0618	(+76.6)	0.0720	(+105.7)	0.0484	(+38.3)	0.0856	(+144.6)
90	0.0181	0.0379	(+109.4)	0.0494	(+172.9)	0.0291	(+60.8)	0.0643	(+255.2)
100	0.0161	0.0241	(+49.7)	0.0337	(+109.3)	0.0081	(-49.7)	0.0378	(+134.8)
avg	0.1442	0.1796	(+24.5)	0.1852	(+28.4)	0.1604	(+11.2)	0.2001	(+38.8)

Table 2: Interpolated precision at 11 recall points for the parts of the the query. Table 1 compares other runs.

above. At 0% recall, the poor low-recall performance of local context analysis caused the combined query to be worse than basic query processing. These results suggest that the weighting used ($1 \times$ basic query processing, $2 \times$ core query processing, $4 \times$ local context analysis) was the opposite of what was required. Further tests have confirmed that LCA should not have been so heavily weighted, though the impact of “better” weights is not large.

In considering possible weightings of the three components, we also determined the best weighting on a per-query basis and ran that. We found a 27.7% improvement in average precision (from 0.2001 to 0.2555) when doing that. Such an improvement is unlikely to be found outside a retrospective analysis, but it does suggest that better weight balancing would be helpful in general.

7 Spanish Results and Discussion

Two sets of results, SIN300 and SIN301, were evaluated in the Spanish track. Both sets were based on automatic processing of TREC topics SP51-SP75 into queries and automatic query expansion. The official results are summarized below.

Query Type	Average Precision			
	5 docs	30 docs	100 docs	11-Pt Avg
SIN300	0.6720	0.5253	0.4044	0.4551
SIN301	0.6800 (+ 1.2)	0.5893 (+ 12.2)	0.4532 (+ 12.1)	0.5058 (+ 11.1)

SIN301 was significantly better above the 5 document cutoff. Recall that the primary difference between the query sets is that SIN300 was based only on topic descriptions, while SIN301 was based on both descriptions and narratives. The narratives contained good concepts which alone improved results by 14%. The combination of global and local analysis by Local Context Analysis is also effective and yields improvements of an additional 15%. This approach is significantly better than last year’s use of InFinder.

8 Chinese Results and Discussion

Precision and recall tables for Chinese experiments follow. The queries were evaluated on the combined Xinhua and Peoples Daily Datasets from TREC-5 (1996). As the results indicate, query expansion makes a significant positive contribution to the quality of retrieval. Stopword removal had little impact.

Regrettably, the original INQUERY submission for Chinese TREC evaluation was incorrect because the query set was run on only the Xinhua text. Missing from the database on the official run was the Peoples Daily text which represents almost 4/5 of the total collection!

Interpolated Recall-Precision Averages				
	Unstopped	Stopped	Expanded 0.5 vs 1 (HIN300)	Expanded 1 vs 1 (HIN301)
at 0.00	0.6905	0.6911	0.7421	0.7490
at 0.10	0.5462	0.5682	0.6125	0.6069
at 0.20	0.4905	0.4924	0.5557	0.5618
at 0.30	0.4304	0.4314	0.4754	0.4849
at 0.40	0.3716	0.3721	0.4272	0.4425
at 0.50	0.3327	0.3308	0.3565	0.3626
at 0.60	0.2607	0.2657	0.2958	0.3109
at 0.70	0.1955	0.1990	0.2387	0.2598
at 0.80	0.1635	0.1660	0.1951	0.2143
at 0.90	0.1069	0.1104	0.1288	0.1422
at 1.00	0.0246	0.0248	0.0301	0.0439
Average	0.3139	0.3170	0.3523	0.3651
Change:		1.0%	12.2%	16.3%
Precision				
At 10 docs:	0.4526	0.4579	0.4737	0.4789
At 100 docs:	0.2558	0.2579	0.2800	0.2905
At 1000 docs:	0.0648	0.0648	0.0679	0.0687
R-Precision	0.3392	0.3380	0.3737	0.3803

9 Routing Results and Discussion

The following table summarizes the results of our routing run:

Set	AvgPrec	R-Prec	Prec@10	Prec@100
45 topics	0.3359	0.3549	0.5156	0.3269
39 topics	0.3633	0.3966	0.5872	0.3756

The "45 topics" are the 45 judged topics which had at least one relevant document. The "39 topics" are the 39 of those which had more than 5 relevant documents.

The following table shows the effect of different parts of the processing. These figures vary from the reported numbers above because they include all 45 topics that had relevant documents, and because the analysis was done with a slightly modified version of Inquiry.

	Base	+terms	+#1	+#uw5	+#uw20	+#uw50	+DFO
Total number of documents over all queries							
Rel_ret:	3665	3860	3864	3867	3867	3872	3998
Interpolated Recall - Precision Averages:							
at 0.00	0.6476	0.7412	0.7331	0.7566	0.7577	0.7564	0.8001
at 0.10	0.5116	0.5687	0.5835	0.5901	0.5916	0.5961	0.6540
at 0.20	0.4134	0.4594	0.4650	0.4702	0.4750	0.4777	0.5271
at 0.30	0.3531	0.3937	0.4007	0.4050	0.4088	0.4114	0.4662
at 0.40	0.2996	0.3323	0.3409	0.3445	0.3482	0.3523	0.3974
at 0.50	0.2579	0.2907	0.2986	0.3023	0.3060	0.3081	0.3504
at 0.60	0.1845	0.2158	0.2174	0.2171	0.2184	0.2225	0.2529
at 0.70	0.1123	0.1442	0.1425	0.1400	0.1421	0.1437	0.1805
at 0.80	0.0809	0.0901	0.0908	0.0942	0.0950	0.0966	0.1285
at 0.90	0.0543	0.0592	0.0590	0.0644	0.0643	0.0621	0.0876
at 1.00	0.0380	0.0421	0.0421	0.0418	0.0418	0.0418	0.0494
Average precision (non-interpolated) over all rel docs							
	0.2502	0.2825	0.2861	0.2891	0.2912	0.2932	0.3333
% Change:		12.9	14.4	15.6	16.4	17.2	33.2
Precision:							
At 5 docs:	0.4622	0.4800	0.4800	0.4889	0.4844	0.4800	0.5111
At 10 docs:	0.4178	0.4511	0.4511	0.4467	0.4511	0.4467	0.4822
At 15 docs:	0.3867	0.4104	0.4148	0.4207	0.4193	0.4207	0.4578
At 20 docs:	0.3611	0.3889	0.3878	0.3933	0.3911	0.3944	0.4344
At 30 docs:	0.3244	0.3511	0.3548	0.3533	0.3630	0.3630	0.3993
At 100 docs:	0.2527	0.2751	0.2760	0.2778	0.2798	0.2804	0.2969
At 200 docs:	0.1944	0.2103	0.2113	0.2116	0.2123	0.2144	0.2246
At 500 docs:	0.1232	0.1314	0.1312	0.1315	0.1319	0.1322	0.1370
At 1000 docs:	0.0814	0.0858	0.0859	0.0859	0.0859	0.0860	0.0888
R-Precision (precision after R (= num.rel for a query) docs retrieved):							
Exact:	0.2816	0.3114	0.3143	0.3260	0.3272	0.3300	0.3677

The results show that expanding the query by 50 terms (“+terms”) yielded a 13% gain. Expansion by pairs of terms (the next four columns) yielded a modest amount, but nothing of substance. The feature reweighing step (Dynamic Feedback Optimization) yielded another healthy improvement. For TREC-5, term expansion and feature reweighing were the key elements.

In TREC-4, we found roughly a 10% improvement from expanding with pairs of terms, so were surprised they they provided so little benefit this year. We have discovered that our term-pair selection code is errorful and are investigating the problem: the selected pairs not only provide minimal improvement on the test collection; they provided a modest *drop* in effectiveness on the training data!

10 Filtering Results and Discussion

Statistics about our results are summarized in the following table:

	Prec	Bal	Rec
Avg pool util	-12.2	1.9	86.0
Avg sampled util	13.9	33.5	159.9
Total retrieved	1276	2844	4623
Total relevant	5808	5808	5808
Total rel ret	807	1469	2209
Avg prec	0.4417	0.4793	0.4843
Avg recall	0.0667	0.1332	0.2466

These results are not particularly pleasing on the surface, but further analysis indicates that the results were relatively good compared to other systems (matching the best reported performance in 40% of the cases). Given that we applied such a simple technique, we are hopeful that further work will improve the utility numbers.

We are intrigued that our high-recall run achieved a higher average precision than our high-precision run but have not yet done the analysis to understand what happened.

11 Summary and Conclusions

Our results this year are very encouraging, not only because they continue to support our approaches, but because they have presented several puzzles for investigating. The conclusions we draw from this work are:

1. Our Local Context Analysis “automatic feedback” technique is consistently helpful across queries, across databases, and across languages.
2. Our techniques for locating the “core concepts” in a query are helpful, though there are several questions their use raises.
3. Combining multiple forms of queries continues to be a successful approach.
4. Longer queries—for example, those including “narrative” prose—once again show themselves to be more useful for retrieving relevant documents.
5. The query expansion and reweighting techniques applied for routing in the past continue to work with new queries and a new routing collection.
6. All of our approaches are successful in languages other than English, and can be applied with surprisingly little difficulty.

We are particularly intrigued by the odd results we found identifying core concepts in the Ad-hoc track, and by the unusual relationship between precision and recall we encountered in the Filtering track. We are pleased that TREC-5 reaffirmed our beliefs, but much more excited by the new areas of exploration raised.

Acknowledgments

Thank you to Bill Curtis for his help with with some of the test runs and Victor Lavrenko for help with results analysis.

This paper is based on research supported by the NSF Center for Intelligent Information Retrieval at the University of Massachusetts, Amherst. This material is also based on work supported

in part by the National Science Foundation, Library of Congress and Department of Commerce under cooperative agreement number EEC-9209623. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

References

- [1] James Allan, Lisa Ballesteros, James P. Callan, W. Bruce Croft, and Zhihong Lu. Recent experiments with INQUERY. In *Fourth Text REtrieval Conference (TREC-4)*, 1995. Forthcoming.
- [2] Chris Buckley and Gerard Salton. Optimization of relevance feedback weights. In Edward A. Fox, Peter Ingwersen, and Raya Fidel, editors, *Proceedings of the 18th annual international ACM SIGIR conference on research and development in information retrieval*, pages 351–357, Seattle, Washington, July 1995. ACM.
- [3] Chris Buckley, Amit Singhal, and Mandar Mitra. Using query zoning and correlation within SMART : Trec 5. In D. Harman, editor, *Proceedings of the Fifth Text REtrieval Conference (TREC-5)*. National Institute of Standards and Technology, 1996.
- [4] J. P. Callan, W. B. Croft, and S. M. Harding. The INQUERY retrieval system. In *Proceedings of the Third International Conference on Database and Expert Systems Applications*, pages 78–83, Valencia, Spain, 1992. Springer-Verlag.
- [5] W. B. Croft, J. Callan, and J. Broglio. TREC-2 routing and ad-hoc retrieval evaluation using the INQUERY system. In D. Harman, editor, *Proceedings of the Second Text REtrieval Conference (TREC-2)*. National Institute of Standards and Technology Special Publication 500-215, 1994.
- [6] W. Bruce Croft and Jinxi Xu. Corpus-specific stemming using word form co-occurrence. In *Symposium on document analysis and information retrieval (SDAIR '95)*, 1995.
- [7] K.L.Kwok. A new method of weighting query terms for ad-hoc retrieval. In *Proceedings of the 19th annual international ACM SIGIR conference on research and development in information retrieval*, pages 187–195, Zurich, 1996. Association for Computing Machinery.
- [8] Robert Krovetz. *Word Sense Disambiguation for Large Text Databases*. PhD thesis, University of Massachusetts, Amherst, 1995.
- [9] J. Ponte and W. B. Croft. USeg: A retrargetable word segmentation procedure for information retrieval. In *Symposium on document analysis and information retrieval (SDAIR '96)*, 1996.
- [10] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In D. K. Harman, editor, *The Third Text REtrieval Conference (TREC-3)*, Gaithersburg, MD, 1995. National Institute of Standards and Technology, Special Publication 500-225.
- [11] Howard Turtle and W. Bruce Croft. Inference networks for document retrieval. In Jean-Luc Vidick, editor, *Proceedings of the 13th International Conference on Research and Development in Information Retrieval*, pages 1–24. ACM, September 1990.

- [12] Howard R. Turtle and W. Bruce Croft. Efficient probabilistic inference for text retrieval. In *RIAO 3 Conference Proceedings*, pages 644–661, Barcelona, Spain, April 1991.
- [13] J. Xu, J. Broglio, and W. B. Croft. The design and implementation of a part of speech tagger for english. Technical Report IR-52, Center for Intelligent Information Retrieval, Department of Computer Science, University of Massachusetts, 1994.
- [14] J. Xu and W. B. Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th annual international ACM SIGIR conference on research and development in information retrieval*, pages 4–11, Zurich, 1996. Association for Computing Machinery.

TREC-5 English and Chinese Retrieval Experiments using PIRCS

K.L. Kwok & L. Grunfeld

Computer Science Dept., Queens College, CUNY,
Flushing, NY 11367. email: kklqc@cunyvm.cuny.edu

ABSTRACT

Two English automatic ad-hoc runs have been submitted: pircsAAS uses short and pircsAAL employs long topics. Our new avtf*ildf term weighting was used for short queries. 2-stage retrieval were performed. Both automatic runs are much better than the overall automatic average. Two manual runs are based on short topics: pircsAM1 employs double weighting for user-selected query terms and pircsAM2 additionally extends these queries with new terms chosen manually. They perform about average compared with the the overall manual runs.

Our two Chinese automatic ad-hoc runs are: pircsCw, using short-word segmentation for Chinese texts and pircsCwc, which additionally includes single characters. Both runs are much better than average, but pircsCwc has a slight edge over pircsCw.

In routing a genetic algorithm is used to select suitable subsets of relevant documents for training queries. Out of an initial random population of 15, the best subset (based on average precision) was employed to train the routing queries for the pircsRG0 run. This ith (= 0) population is operated by a probabilistic reproduction and crossover strategy to produce the (i+1)th and evaluated, and this iterates for 6 generations. The best relevant subset of the 6th is used for training our queries for pircsRG6. It performs a few percent better than pircsRG0, and both are well above average.

For the filtering experiment, we use thresholding on the retrieval status values; thresholds are trained based on the utility functions. Results are also good.

1. ENGLISH AD-HOC RETRIEVAL

1.1 Automatic Query Formulation

TREC-5 ad-hoc retrieval requires experimenting with

a short form of the topics (#251-300) having only a description section, and a long form that has additionally a narrative section. The collection consists of Tipster Disk 2 (WSJ, AP, ZIFF, FR) and Disk 4 (FR, CR-E, CR-H, FT). Our experiments are designated as pircsAAS and pircsAAL. When query descriptions are short (in this case averaging about 6 content terms per query for pircsAAS - see Fig.1), the users intention becomes vague and ambiguous, and retrieval is notoriously difficult. Short queries lack both term variety (to describe the intended topics) and

Number of Uniq.Terms	Number of Queries			
	AAS	AAL	AM1	AM2
1	1		1	
2	2		4	
3	6		6	2
4	5		8	2
5	7		8	3
6	9	1	5	8
7	6		5	7
8	6	1	6	6
9	1		1	2
10	4	2	3	4
11	1	3	1	4
12		2		5
13		1		3
14		1		
15	1	4	1	1
16		3		
17		1		1
18		1		1
19	1	1	1	
20		1		
21-42		27		1
74		1		
Average	6.32	22.02	5.92	9.00

Fig.1: Distribution of Query Sizes for pircsAAS, AAL and pircsAM1, AM2.

term weight indicating importance (because practically all terms are used once). A comparison of query lengths for this set of queries (#251-#300) is given in Fig.1. It is noted that the long-form queries in pircsAAL have nearly 4 times as much text description as the short-form pircsAAS. We recently introduced a method of automatically weighting terms in short queries that appears to enhance the previous TREC-4 retrieval results [Kwok96], and have applied it in TREC-5. Each query term is assigned a weight given by:

$$w = \text{avtf}^\alpha * \text{idf} / \text{norm}$$

where

avtf = average term frequency

= ColFre/DocFre

idf = inverse log document frequency
with cutoff

= $1/\log \max(\text{cutoff}, \text{DocFre})$

α = a parameter set to 1.5

norm = a normalizing factor, so that
the total term weights for a
document sum to 1.

This is a form of 'tf*idf' weight popularized by Salton. The average term frequency avtf is the default within-document term frequency based on observation from the whole collection, and given to a query term when the short query does not provide differentiating clues. The inverse log document frequency idf (which was called idf in [Kwok96] and used successfully in our routing experiments) attaches less weight to terms with high document frequency, but not so much less as using the idf. At the low frequency end, a cutoff value of 2000 prevents low frequency terms from getting unusually high weights. α is a parameter that allows one to vary the importance of the two factors: avtf and idf.

We also add term variety to a query in a small scale. Normally terms above a frequency threshold HiFre (set to 60000 in our case) are removed as stopwords. We believe that they still can carry some content, especially in conjunction with other query terms. In TREC-5, we automatically concatenate them with other less frequent query terms to form 2-word phrases, and these have a chance to match with 2-word adjacency phrases that are captured in documents. This experiment is labeled Ad-hoc Automatic Short: pircsAAS.

For long queries we suppress the term variety addition. The avtf*idf term weight is still applied to those queries with 14 terms or less; otherwise our default weighting using actual observed term frequencies is performed. This experiment is labeled pircsAAL.

Retrieval is done as before in 2-stages [BSAS95, RWJHG95, KwGL95]. The initial stage uses the raw query enhanced by weighting and limited term variety modifications as discussed above, and a ranked list of documents is obtained. The 40 best-ranked subdocuments are then used for feedback and query expansion with 50 terms. The second stage retrieval uses this expanded query to produce the final retrieval ranked list.

1.2 Manual Query Formulation

Manual queries are formed using the short-form topics as basis as in [KwGr96]. In pircsAM1 (Ad-hoc Manual 1), we double weight certain judged query terms as in TREC4 and the operation is simple and not burdensome, averaging 1/2 minute per query. In pircsAM2 we add some associated terms that either come from the narrative section or based on the opinion of the investigator and takes about 2-3 minutes per query. Table 1 also tabulates their query size distributions. pircsAM1 is shorter than pircsAAS because no manual new terms or automatic term pairs are added. pircsAM2 queries have average length of 9. We manually added about 3 terms to pircsAM1 on average. Retrieval is done similarly as before. This will provide a comparison of our automatic query formulation with respect to manual within our PIRCS system.

1.3 Ad-hoc Results and Discussion

The collection on Disk 2&4 are divided into four subcollections with documents segmented into about 550-word chunks ending on a paragraph boundary as in previous TRECs. They are served by a master lexicon of 664,064 unique terms that include 80,031 entries of our semi-automatic 2-word adjacency phrases. This is about 50% more than our previous phrase list of 55,599.

Comparison Within Pircs

In Fig.2 we have tabulated our 4 official ad-hoc retrieval results (shown *) together with their

QryType pircs	1st-stage Retrieval	2nd-stage QryExp Retrieval
=====		
AUTOMATIC		
AAO	1763/.1397 (%/%) (% / %)	2279/.1612 (29.3 / 15.3) (% / %)
AAS	1990/.1605 (%/%) (12.9 / 14.9)	*2335/.1809 (17.3 / 12.7) (2.5 / 12.3)
AAL	2452/.2082 (%/%) (39.1 / 49.0)	*2941/.2466 (19.9 / 18.4) (29.0 / 53.1)
MANUAL		
AM1	1970/.1610 (%/%) (11.7 / 15.2)	*2354/.1856 (19.5 / 15.3) (3.3 / 15.2)
AM2	2375/.2033 (%/%) (34.7 / 45.5)	*2801/.2298 (17.9 / 13.0) (22.9 / 42.6)

Fig.2: Ad-Hoc Experiments: Relevants Retrieved @1000 / Non-Interpolated Average Precision Results Before and After Query Expansion for the 5 Initial Query Types Averaged over 50 queries.

corresponding 1st-stage retrievals for comparison. Additionally, another pair of experiments labeled **pircsAAO** is also shown. This is the Ad-hoc Automatic Original retrieval using the raw short queries and **pircs'** default weighting (ie. not using **avtf*ildf** or term variety additions). Percentage increases from this base values (%/%) are shown both horizontally (before and after query training and expansion) and vertically (between query types).

The results confirm what has been reported previously in [Kwok96] where the experiments were done with known answers and not blind, and in TREC-4 [KwGr96]. In particular, the 2-stage expanded query retrieval outperforms single stage retrieval on average in all cases, by between 12.7% to 18.4% in precision and 17.3% to 29.3% in relevants retrieved. It is noted that retrieval by the raw short queries with default weighting (**pircsAAO**) is not recommended. The queries should either be automatically weighted via the **avtf*ildf** scheme (**pircsAAS**, 12.3% to 14.9% improvement in precision compared to **pircsAAO**), manually double weighting some user-selected important terms (**pircsAM1**, 15.2% improvement), or additionally manually add some associated content

pircs	Number of Queries		
	Better	Equal	Worse
=====			
AAS (compared to short topic runs)			
Av.P	32(3)	3	15
rel.ret@100	33(8)	6	11
rel.ret@1000	35(10)	6	9
AAL (compared to long topic runs)			
Av.P	38(11)	9	3
rel.ret@100	36(19)	8	6
rel.ret@1000	41(26)	4	5
AM1 (compared to manual runs)			
Av.P	22(3)	5	23
rel.ret@100	18(8)	7	25
rel.ret@1000	26(8)	2	22
AM2 (compared to manual runs)			
Av.P	22(1)	3	25
rel.ret@100	25(8)	7	18
rel.ret@1000	37(13)	1	12

(.) denotes number equally the best results.

Fig.3: Comparison of Ad-Hoc Results with the Median from All Sites

terms (**pircsAM2**, over 42.6% to 45.5% improvement). The automatic weighting results for short queries **pircsAAS** are quite comparable to the manual double-weighting method **pircsAM1**. However, the best performance is obtained with the original long queries that include the narrative section (**pircsAAL**, about 49% to 53.1% improvement). Apparently the collective set of terms (average 22.07) provided by the topic composer gives clues to the retrieval system that are lacking in short descriptions.

Also, adding term variety in initial retrieval based on our simple procedure of paring high frequency terms (that are removed) with existing low/medium frequency terms is not recommended. Removing this operation actually improves our **pircsAAS** results by about 6% in 2nd stage retrieval.

Comparison with Other Sites

Fig.3 shows how our results compare with respect to the median from all sites. The automatic runs, whether short or long, perform very well. However, the manual runs are about median.

As in previous years we have also calculated the MAXI-system performance by selecting the best results among all sites for each of the 50 queries. Three measures (average precision, relevants retrieved @100 and @1000) are used. Fig.4 shows the comparison of our system to this performance. Our automatic results for long queries achieve between 78.9% to 86.7% of these maximum values. Short query results pircsAAS, like manual pircsAM2, achieve about 54% to 70% of their respective maxima. PircsAM1 performed worse. It is interesting to note that the MAXI-system for short queries actually outperformed that for long queries in two measures: average precision and relevants retrieved @100. Our simple manual manipulation of the queries are not sufficient to achieve good results compared with what are done in other sites.

	Av.P.	RR@100	RR@1000
=====			
MAXI (short)	.3312	1413	3338
pircsAAS	.1809 (54.6%)	864 61.1%	2335 70.0%)
MAXI (long)	.3124	1343	3392
pircsAAL	.2466 (78.9%)	1069 79.6%	2941 86.7%)
MAXI (manual)	.4371	1909	4023
pircsAM1	.1856 (42.5%)	848 44.4%	2354 58.5%)
pircsAM2	.2298 (52.6%)	1011 53.0%	2801 69.6%)

Fig.4: Comparison of PIRCS's Performance to the MAXI-System

2. CHINESE AD-HOC RETRIEVAL

2.1 Word Segmentation

As is well known, a Chinese written sentence consists of a string of characters between punctuations and without natural word boundaries. A crucial step to Chinese IR is to segment a sentence into meaningful content words. This operation is hard because different adjacent character sequences can be meaningful depending on context, and it is not obvious which sequence to choose. It is usual to consider the longest meaningful character sequence breakups as the correct segmentation of a sentence. For large scale IR however, this may not be fruitful because long words can lead to problems of evaluating partial matchings between query and document words. A number of studies and operational systems for Chinese word segmentation has been done including: [WuTs95,Chie95,LiLY96,PoCr96,NMSU].

Our strategy is to detect common short words of 2 or 3 characters long (with some proper names of 4 characters also), and use exact matching for retrieval. Our purpose is to generate useful features for retrieval, and not necessarily for human consumption. Readability most probably implies good features, but not necessary vice versa. To achieve segmentation for a sentence in a relatively efficient manner, we implement a four-step procedure with a mixture of expert knowledge and statistical processing:

1) facts - lookup on a manually created lexicon list of about 2000 items. Each item is tagged as useful, useless (stopword), numeric, punctuation and a few other codes. Longest match is performed when searching on this lexicon, and this results in breaking a sentence into smaller chunks of texts. This procedure ignores other possible breakup paths. However, we conjecture that its effect may not be too significant because quite often the ambiguities are with character sequences that form stopwords anyway, and also the topics may be sufficiently long that these errors may be remedied in other parts of the query.

2) rules - common usage ad-hoc rules, also manually determined, are then employed to further split the chunks into short words of 2 or 3 characters. Examples of rules that serves to segment text are: a) any two adjacent similar characters XX; b) AX, where A is a small set of special characters; c) a remaining sequence of even number of characters are segmented two by two.

3) filtering - a first pass through the test corpus using steps 1 and 2, and frequency of occurrence is used to threshold the captured short words to extract the most common occurring words.

4) iteration - expand the initial lexicon list in step 1) based on step 3) to about 15000 in our case and re-process the corpus. No additional training collection is needed in these four steps.

Our procedure is like data-mining common short words in the domain of the collection. Naturally no segmentation algorithm is perfect and our case is no exception. Wrong segmentations are made, but for IR we hope the error rates may be tolerable. We evaluate our result with a manual short-word segmentation on the topics and found that we achieve 91.7% in recall and 84.3% in precision. Some examples of our segmentation result with all stopwords kept are given in the Appendix.

The words detected in the above 4-step procedure are used for document and query representation directly in pircsCw. For pircsCwc, we further break up each word into characters and use both words and characters for representation. Because we use exact matching for retrieval, pircsCwc may retrieve documents that may have been missed by pircsCw. 2-stage retrieval is done similar to Section 1.1. However, a more conservative query expansion is done with 30 terms from the best 40 ranking documents of the 1st stage retrieval.

2.2 Ad-hoc Chinese Retrieval Results

We also segment our long Chinese documents into subdocuments of 550 characters long chunks ending on a paragraph boundary as in English. The initial 24,988 Xinhua and 139,801 People’s Daily collections produce a total of 231,527 subdocuments. Using our word segmentation procedure, pircsCw processing accumulates a lexicon list of 492,293 entries. With pircsCwc where single characters from identified words are also used for representation, the lexicon size becomes 494,288.

Comparison Within Pircs

Fig.5 tabulates our 2 official Chinese ad-hoc retrieval results (shown *) together with their corresponding 1st-stage retrievals for comparison. Percentage

QryType	1st-stage	2nd-stage QryExp
pircs	Retrieval	Retrieval
=====		
AUTOMATIC		
Cw	1858/3548 (%/%) (% / %)	*1997/3989 (7.5 / 12.4) (% / %)
Cwc	1925/3726 (%/%) (3.6 / 5.0)	*2019/4226 (4.9 / 13.4) (1.1 / 5.9)

Fig.5: Chinese Ad-Hoc Experiment: Relevants Retrieved @1000 / Non-Interpolated Average Precision Results Before and After Query Expansion for the 2 Initial Query Types Averaged over 28 queries.

increases from base values (%/%) are shown horizontally (before and after query training and expansion) and vertically (between query types). Results shows that the 2-stage expanded query retrieval works also in the Chinese environment, outperforming initial stage retrieval in both cases, by between 12.4% to 13.4% in precision and 4.9% to 7.5% in relevants retrieved. Using characters in addition to short words for representation in pircsCwc has a slight edge of a few perecent over pircsCw which uses short words alone. pircsCwc only outperforms pircsCw in exactly half of the topics, but the amount is larger. For example, in queries #14, 17, 22, 25, pircsCwc returns average precisions that are better than pircsCw by over 0.15. The reverse is true only on one occasion with query #23.

(a)	死亡	related to	死, 死伤
	character shared: 死		
(b)	灭绝	related to	灭种
	character shared: 灭		
(c)	爱滋	unofficial form of	艾滋
	character shared: 滋		

We can see why use of characters can help achieve better performance in some circumstances. In the box above, example (a) is a word from query #22 meaning

'dead' and can be expressed in related forms as shown. Another example is (b) meaning 'extinct' query #25. If perfect segmentation is performed but alternative forms are used in the query and a document, there will be no matching value between query and document even though they talk about the same concept. However, if the chinese words are broken up into single characters, some character(s) are shared and non-zero matching values result. Chinese characters do carry meaning though often imprecise; they probably lie between alphabets and words in this capacity.

A different situation occurs in topic #14 and shown in example (c) for the different transliteration of 'AIDS'. Again correct word segmentation leads to no matching with many documents that use the official form. However, when single characters are used, the second character happens to be the same in both transliterations and some matching between query and relevant documents is restored. Of course one also has to balance this with wrong matchings due to characters being used in multiple senses.

Comparison with Other Sites

Fig.6 shows that our two official runs perform very well when compared with the median from all sites. In particular, pircsCwc returns the best average precision of 0.4226, and out of 28 queries, 25 queries are better and 1 equals the medium, while 2 are worse. Out of these 25, 8 returns the best.

	Number of Queries		
pircs	Better	Equal	Worse
=====			
Cw			
Av.P	22(6)	1	5
rel.ret@100	22(8)	3	3
rel.ret@1000	27(16)	0	1
Cwc			
Av.P	25(8)	1	2
rel.ret@100	23(10)	3	2
rel.ret@1000	27(19)	0	1

Fig.6: Comparison of Chinese Ad-Hoc Results with the Median from All Sites

3. ROUTING EXPERIMENTS

Training Document Selection

Retrieval results can be improved by an intelligent selection of training documents. We found [KwGr94] that short documents and high ranking subdocuments are particularly effective. Short documents can be extracted without the need for a retrieval and they include information that would not be found by the retrieval system. High ranking documents contain information that the system considers close to the query concept.

Selection and weighing of training data became subject of current research in machine learning. Breiman [Brei96] discovered, that bagging improves the performance of CART, a decision tree learning procedure. During bagging the training set is sampled with replacement, thus perturbing the resulting decision tree. After a number of trials the results are added together using a voting procedure. Breiman observes that this procedure is effective for an unstable learning method, where a small change in training data can have a major effect on the classifier.

Schapire et al. [FrSc96] proposed another method called boosting. In boosting the misclassified training cases are given additional weight. This procedure is iterated until there is no more improvement. Quinlan [Quin96] applied both of these methods to the C4.5 decision tree system and found significant improvement for both methods.

Preliminary experiments revealed, that bagging was not helpful for the pircs system as expected, since the pircs network model is based on bayesian statistical retrieval, thus it is a stable learning method. Boosting was beneficial in some cases, but in most cases it was not. However we found that applying a genetic algorithm search to the document selection problem was beneficial.

Genetic Algorithms and IR

Genetic algorithms [Gold89] is a search procedure based on natural selection, the survival of the fittest. The population consists of strings of data of some alphabet, often consisting 1's and 0's, they may be identified as genes. The initial population is generated randomly and the fitness of each individual is

calculated The strings reproduce, creating the next generation. The most common operations used for reproduction are selection, crossover and mutation. At the time of selection higher ranking strings have a greater chance of being selected, this is how the fitness function guides the search. Crossover is the process of combining two strings into one. Mutation causes random changes to the genes. The new population selected forms the next generation. New generations are formed until the fitness stops improving or some other stopping criteria is reached. Genetic algorithms (GA) have been applied to the IR relevance feedback domain by a number of researchers [Chen95] [YaKoRa93].

The algorithm can be summarized as follows [Gre88]:

```

Procedure GA
begin
  initialize population P(0)
  evaluate P(0)
  t=1
  repeat
    select P(t) from P(t-1)
    recombine P(t)
    evaluate P(t)
  until (termination condition)
end

```

Genetic algorithms are known to perform well in many complicated search spaces, however their computational cost is greater then the cost of other common search methods.

When applying GA search to the IR problem the strings may represent the presence or absence of terms, in which case the result is the best subset of terms for the query. Or it can be an encoding of weights for each term. An interesting variation is when they represent Boolean connectors and terms, then the result is a Boolean query [KPBS94]. In our experiment the goal is to find the best training subset, therefore the string represented presence or absence of training documents from the full set for the query. The population was initialized randomly and it also included the string representing all documents. The fitness function used was the TREC evaluation package. The training documents were the top 3000 short (less than 550 words) evaluated documents.

Genetic algorithms often require hundreds or even

thousands of generations to reach their maximum fitness. In our case since a complete training and retrieval is required for the fitness function, we limited the number of generations to six and the population to 15. Mutation was not used. To compensate for the limited number of evaluations, we used an elitist selection strategy giving high ranking individuals very high probability of selection. In genetic algorithms this often leads to gene loss and premature convergence to a local maximum. Even with this strategy the fitness value reached at training continued to increase and the limit of six generations was imposed due to time constraints.

	base	gen 0	gen 6
avg	0.3233	0.3142	0.3402
%chg		-2.81%	5.26%

Fig.7: Comparison of Routing Queries to Baseline.

gen	0	1	2	3	4	5	6
avg	.3177	.3343	.3255	.324	.3369	.3389	.3399
%chg		5.23	2.46	1.98	6.04	6.67	6.99

Fig.8 Performance of the best member in each generation.

	pircsRG0			pircsRG6		
	>	=	<	>	=	<
=====						
av. prec:	38(3)	3	4	41(3)	3	1
rel_ret						
@ 100:	34(9)	9(2)	2	39(11)	6(2)	0
rel_ret						
@ 1000:	34(15)	6(5)	1	38(18)	7(5)	0

(figure in paranthesis is number of queries equaling the best values)

Fig.9: Comparison of Routing Results with Median.

The standard crossover operation used in a GA consist of cutting the string of the two parents at a random point and exchanging one part to create the new offsprings. In our scheme each gene was added probabilistically. Thus if both parents had a 1 in the same position the offspring could have a 0, 1 or 2.

This was done to allow for training documents to have weight.

Two routing queries were submitted. PircsRG0 is the best member of generation 0, pircsRG6 is the best of generation 6. In our tests they both outperformed the baseline query, which was similar to our pircsL query submitted for TREC-4.

3.1 ROUTING RESULTS

Generation 6 improved on the baseline measure but generation 0 did not. See Fig.7.

A closer look at the results however reveals that most of the gain was in one query. [q 207] . The gain for the 39 queries that have 6 or more relevant documents is only 1.2%.

	Precision oriented =====	Balanced =====	Recall oriented =====
Relevant	5808	5808	5808
Retrieved	861	1703	3056
Rel. Retr.	576	977	1465
Mean Prec.	.215	.246	.423
Mean Recall	.043	.091	.162
Total Prec.	.669	.574	.479
Total Recall	.099	.168	.252
Total Score	-279	251	2804

Fig.10: Filtering results.

	Number of Queries		
	>	=	<
	=====		
Precision oriented	13(5)	28(15)	8
Balanced	24(12)	18(12)	7
Recall oriented	28(11)	10(3)	11

(figure in paranthesis is number of queries equaling the best values)

Fig.11: Comparison of Filtering Results with Median.

Fig.8 shows that there is an upward trend as the generations advance. These runs were run with query expansion factor of 180 as opposed to 300 on the official run. Fig.9 compares our routing results to the median from all sites.

4. FILTERING EXPERIMENTS

The filtering track experiment used the same approach that we used for TREC-4. We find the point where the maximum occurs for the utility function during a retrospective retrieval. If there is more than one maxima we use the later one. The RSV for this point is used for the cutoff value on the test collection. The cutoff values were extrapolated from a retrieval on the WSJ and AP collections from disk 2. Since that was a retrospective retrieval the cutoff points may have been too conservative. In the future we will consider other possible predictors too, such as rank and number of relevants in the training document. It is also possible that using more data than just disk2 will result in more accurate prediction. The results are summarized in the Fig.10 and 11.

5. CONCLUSION

Our PIRCS system has been extended to handle Chinese documents (GuoBiao coded). We have also implemented a Chinese word segmentation algorithm for IR. Otherwise, processing is similar as in English. It appears that except for manual ad-hoc runs which we did not expand too much effort, all the automatic ad-hoc (including Chinese), routing and filtering runs are much better than the overall average from all sites. We attribute this to the highly effective PIRCS retrieval engine and its learning capabilities which have consistently been demonstrated to give exemplary performance.

ACKNOWLEDGMENTS

This work was partially supported by a PSC-CUNY grant and a Tipster program grant from DARPA . The following are especially acknowledged for their effort in developing the Chinese dictionary and part of the programs for Chinese processing: Yen Yen Lee, Eric Rong, Jun Wang and Mark Young.

REFERENCES

- [BSAS95] Buckley, C, Salton, G, Allan, J & Singhal (1995). Automatic query expansion using SMART: TREC 3. In: Overview of the Third Text REtrieval Conf. (TREC-3). Harman, D.K.(ed.) NIST SP 500-225; pp.69-80.
- [Brei96] Breiman, L (1996). Bagging predictors, Machine Learning, to be published.
- [Chen95] Chen, Hsinchun (1995). Machine Learning for information Retrieval: Neural Networks, Symbolic Learning, and Genetic Algorithms, JASIS 46(3) 194-216.
- [Chie95] Chien, L.F (1995). Fast and quasi-natural language search for gigabytes of Chinese texts. In: Proc. 18th ACM SIGIR Conf. on R&D in IR. Fox, E., Ingwersen, P. & Fidel, R. (eds.) ACM:NY, NY. pp.112-120.
- [FrSc96] Freund Y, Schapire R,E (1996). A decision theoretic generalization of on-line learning and an application to boosting. 1996 , (["http://www.research.att.com/orgs/ssr/people/schapire"](http://www.research.att.com/orgs/ssr/people/schapire))
- [Gold89] Goldberg, D. E. Genetic Algorithms in Search Optimization & Machine Learning. Addison Wesley 1989.
- [KPBS94] Kraft, D.H, Petry, F.B, Buckles, B.P, Sadasivan, T (1994). The use of genetic Programming to Build Queries for Information Retrieval. IEEE Symposium on Evolutionary Computation, Orlando, 1994.
- [KwGL95] Kwok, K.L., Grunfeld, L. TREC-3 ad-hoc, routing retrieval and thresholding experiments using PIRCS. (1995). In: The Overview of the Third REtrieval Conference TREC-3. Harman, D.K. (ed.) NIST SP 500-225: Gaithersburg, MD. pp.247-255.
- [KwGr94] Kwok K.L, Grunfeld L (1994). Learning from relevant documents in large scale routing retrieval. In Proc ARPA Human Language Technology Workshop, Mar 8-11, 1994 Plainsboro NJ.
- [KwGr96] Kwok, K.L. & Grunfeld, L (1996). TREC-4 ad-hoc, routing retrieval and filtering experiments using PIRCS. In: The Fourth Text REtrieval Conference (TREC-4). Harman, D.K. (Ed.). NIST Special Publication 500-236, pp.145-152.
- [Kwok96] Kwok, K.L (1996). A new method of weighting query terms for ad-hoc retrieval. Proc. 19th Annual Intl. ACM SIGIR Conf. on R&D in IR. pp.187-195.
- [LiLY96] Liang, T., Lee, S.Y & Yang W.P (1996). Optimal weight assignment for a Chinese signature file. Info. Proc. Mgmt. 2:227-237.
- [NMSU] 'Algorithm of Chinese segmenter' from CRL, New Mexico State University.
- [PoCr96] Ponte, J & Croft, W.B (1996). USeg: a retargetable word segmentation procedure for information retrieval. In: Symposium on document analysis and information retrieval (SDAIR '96).
- [Quin96] Quinlan, J.R. Boosting, Bagging and C4.5, Draft of paper to be given at AAAI 96 ("ftp://ftp.cs.su.oz.au/pub/ml/q.aaai96.ps")
- [RWJHG95] Robertson, S.E, Walker, S, Jones, S, Hancock-Beaulieu, M.M & Gatford, M (1995). Okapi at TREC-3. In: The Overview of the Third REtrieval Conference TREC-3. Harman, D.K. (ed.) NIST SP 500-225: Gaithersburg, MD. pp.109-126.
- [WuTs95] Wu, Z & Tseng, G (1995). ACTS: An automatic Chinese text segmentation system for full text retrieval. J. ASIS, 46:83-96.
- [YaKoRa93] Yang, J. Korfhage, R.R. Rasmussen, E (1993). Query improvement in information retrieval using genetic algorithm: a report on the experiments of the TREC project. In Text Retrieval Conference (TREC-1) pp 31-58, Gaithersburg, MD 1993.

Query 002: Original Chinese

中共对于中国统一的立场

中国，一国两制，台湾，和平统一，经贸合作，两岸关系，科技、文化交流

相关文件必须提到中共如何经由实现一国两制来达到台湾与大陆统一的目的。如果文件只是外国政府重申支持中共对台湾拥有主权或提到中共与其他国家之经贸、科技、文化交流，则为不相关文件。

Segmented Result:

中共 | 对于 | 中国 | 统一 | 的 | 立场 | 中国 | 一国 | 两 | 制 |
台湾 | 和平 | 统一 | 经贸 | 合作 | 两 | 岸 | 关系 | 科技 | 、 |
文化 | 交流 | 相关文件 | 必须 | 提到 | 中共 | 如何 | 经由 | 实现 | 一国 |
两 | 制 | 来 | 达 | 到 | 台湾 | 与 | 大陆 | 统一 | 的 |
目的 | 如果 | 文件 | 只 | 是 | 外国 | 政府 | 重申 | 支持 | 中共 |
对 | 台湾 | 拥有 | 主权 | 或 | 提到 | 中共 | 与其 | 他 | 国家 |
之 | 经贸 | 、 | 科技 | 、 | 文化 | 交流 | 则 | 为 | 不 | 相关文件 |

Query 014: Original Chinese

中国的爱滋病例

中国,云南,爱滋病,HIV,高危险群患者,注射器,病毒

相关文件应当包括中国那些地区的爱滋病例最多,爱滋病毒在中国是如何传播的,以及中国政府如何监测爱滋病并控制它的传染。

Segmented Result:

中国 | 的 | 爱滋 | 病例 | 中国 | 云南 | 爱滋病 | ,HIV | 高 | 危险 |
群 | 患者 | 注射 | 器 | 病毒 | 相关文件 | 应 | 当 | 包括 | 中国 |
那些 | 地区 | 的 | 爱滋 | 病例 | 最 | 多 | 爱滋 | 病毒 | 在 |
中国 | 是 | 如何 | 传播 | 的 | 以及 | 中国 | 政府 | 如何 | 监测 |
爱滋病 | 并 | 控制 | 它 | 的 | 传染 |

'^' denotes missing separator; '-' denotes spurious separator

Okapi at TREC-5

M M Beaulieu* M Gatford* Xiangji Huang* S E Robertson* S Walker*
P Williams*

Jan 31 1997

Advisers: E Michael Keen (University of Wales, Aberystwyth), Karen Sparck Jones (Cambridge University), Peter Willett (University of Sheffield)

1 Introduction: summary

General

City submitted two runs each for the automatic ad hoc, very large collection track, automatic routing and Chinese track; and took part in the interactive and filtering tracks. There were no very significant new developments; the same Okapi-style weighting as in TREC-3 and TREC-4 was used this time round, although there were attempts, in the ad hoc and more notably in the Chinese experiments, to extend the weighting to cover searches containing both words and phrases. All submitted runs except for the Chinese incorporated run-time passage determination and searching. The Okapi back-end search engine has been considerably speeded, and a few new functions incorporated. See Section 3.

Automatic ad hoc (and VLC track)

After a good deal of rather fruitless exploration, including further work with two-term phrases, the method finally used was much the same as in TRECs 3 and 4: expansion using terms from the top documents retrieved by a pilot search on topic terms. City96a1 used all topic fields and city96a2 the <desc> field only. The results appear disappointing. Additional runs made since the relevance information became available seem to show that we would have done better without expansion. Two runs using the method of city96a1 were also submitted for the Very Large Collection track. See Section 4.

Automatic routing (and filtering track)

These runs used elaborated versions of City's TREC-3 and 4 methods. The training database and its relevant documents were partitioned into three parts. Working on a pool of terms extracted from the relevant documents for one partition, an iterative procedure added or removed terms and/or varied their weights. After each change in query content or term weights a score was calculated by using the current query to search a second portion of the training database and evaluating the results against the corresponding set of relevant documents. Methods were compared by evaluating queries predictively against the third training partition. Queries from different methods were then merged, and the results evaluated in the same way. Two official runs were submitted, one (city96r2) using the best *method* to produce a set of queries from two halves of the training database, the other (city96r1) using the best query for each topic from those produced during training. These runs appear to have been fairly successful. Three filter track runs and thresholds were also produced using the queries from city96r1 with thresholds derived from the "predictive" portion of the training database. See Section 5.

*Centre for Interactive Systems Research, Department of Information Science, City University, Northampton Square, London EC1V 0HB, UK. email {mmb,mg,xjh,ser,sw,pw}@is.city.ac.uk

Chinese track

City took part in the Chinese track for the first time. Two runs were submitted, one based on character searching and the other on words or phrases (words being identified by lookup as there are no word-delimiters in Chinese). Much of the work involved investigating plausible methods of applying Okapi-style weighting to phrases. (Section 6.)

Interactive

The task definition and the type of users conducting the searches were the main differences for this round of interactive TREC. Past performance in interactive searching had shown that the query expansion facility in the Okapi system was effective in retrieving additional similar items. However, the task of finding different aspects proved to be more taxing. To cater for the different type of users, the GUI used had added functionality to (a) allow users to reverse certain actions taken, and (b) to allow the addition of terms extracted from relevant documents to the query to be handled incrementally rather than requested explicitly by the user. The results were rather disappointing and clearly indicate that we need to carry out further experiments to establish an effective method of conducting such a task. (Section 7.)

2 Background

The search systems City have used for TREC are descendants of the Okapi systems which were developed at the Polytechnic of Central London¹ between 1982 and 1988 under a number of grants from the British Library Research & Development Department and elsewhere. These early Okapi systems were experimental highly-interactive reference retrieval systems of a probabilistic type, some of which featured query expansion [1, 2, 3].

For TREC-1 [4], the low-level search functions were generalized and split off into a separate library — the Okapi Basic Search System (BSS). Interfaces or scripts for batch processing access the BSS using a simple command language-like protocol.

City's TREC-1 results were very poor [4], because the classical Robertson/Sparck Jones weighting model [5] which Okapi systems had always used took no account of document length or within-document term frequency. During TREC-2 and TREC-3 a considerable number of new term weighting and combination functions were tried; a runtime passage determination and searching package was added to the BSS; and new methods of routing query enhancement were developed [6, 7]. Our TREC-3 automatic routing and ad hoc results were relatively good.

TREC-4 [8] did not see any major developments. Routing term selection methods were further improved. In view of the very concise topic statements for the ad hoc, a try was made at manual tweaking of the automatically derived queries; this was a sad failure.

City have always taken part in the Interactive TREC tracks. The TREC-1 system was a simple command-language version of the old pre-TREC Okapi systems. Different interfaces have been written each year, partly in conformance with whatever was the current definition of the interactive tasks. But the interfaces have also become steadily more colourful and GUI-oriented.

3 The system

3.1 The Okapi Basic Search System (BSS)

The BSS, which has been used in all City's TREC experiments, is a set-oriented ranked output system designed primarily for probabilistic-type retrieval of textual material using inverted indexes. There is a family of built-in weighting functions as defined below (equation 1) and described in [7, Section 3]. In addition to weighting and ranking facilities it has the usual boolean and quasi-boolean (positional) operations and a number of non-standard set operations. Indexes are of a fairly conventional inverted type.

The BSS undergoes continual small changes (not always for the better). This time a lot of work was done on trying to speed it up, particularly with regard to set combination, with the result that under reasonable conditions routing term selection runs at about four times the speed it did during TREC-4.

¹Now the University of Westminster.

Weighting functions

The functions available were identical to those used in TRECs 3 and 4. The only ones used during TREC-5 were varieties of the **BM25** function [7, Section 3.2]

$$\sum_{T \in Q} w^{(1)} \frac{(k_1 + 1)tf}{K + tf} \frac{(k_3 + 1)qtf}{k_3 + qtf} + k_2 \cdot |Q| \cdot \frac{avdl - dl}{avdl + dl} \quad (1)$$

where

Q is a query, containing terms T

$w^{(1)}$ is the Robertson/Sparck Jones weight [5] of T in Q

$$\log \frac{(r + 0.5)/(R - r + 0.5)}{(n - r + 0.5)/(N - n - R + r + 0.5)} \quad (2)$$

N is the number of items (documents) in the collection

n is the number of documents containing the term

R is the number of documents known to be relevant to a specific topic

r is the number of relevant documents containing the term

K is $k_1((1 - b) + b \cdot dl/avdl)$

k_1 , b , k_2 and k_3 are parameters which depend on the database and possibly on the nature of the topics. For the TREC-5 experiments, typical values of k_1 , k_3 and b were 1.0–2.0, 8 and 0.6–0.75 resp., and k_2 was zero throughout

tf is the frequency of occurrence of the term within a specific document

qtf is the frequency of the term within the topic from which Q was derived

dl is the document length (arbitrary units)

$avdl$ is the average document length.

When k_2 is zero, as it was for all the TREC-5 experiments except the Chinese, equation 1 may be written in the simpler form

$$\sum_{T \in Q} w^{(1)} \frac{(k_1 + 1)tf}{K + tf} \frac{(k_3 + 1)qtf}{k_3 + qtf} \quad (3)$$

3.2 Hardware

Most of the TREC processing was done on two Suns: an SS20 with 160 MB of memory and an SS10 with 320 MB, both single-processor. Most of the 25 GB of disk was attached to the SS10.

3.3 Databases

Disks 1 and 2 were used for ad hoc “training”; disks 1–3 + the TREC-4 routing data for the routing training. Apart from the official TREC-4 ad hoc and routing test databases, a new database of disks 1–4 was made for the VLC task. The usual three-field structure was used, common to all source datasets, as for all TRECs after the first. The first field was always the DOCNO and the third field contained all the searchable text, mainly the TEXT portions but also headline or title-like material for some datasets and documents. The second field was unindexed (and unsearchable) and so only (possibly) useful for display to users of the interactive system. It was empty except in the case of SJM, when it contained the DESCRIPT field; and the Ziff JOURNAL, AUTHOR and DESCRIPTORS fields. All the databases were set up in such a way that search-time passage determination and searching could be done.

4 Automatic ad hoc

4.1 Outline

Most of the experiments were done using the TREC-3 ad hoc topics (151–200) and database (disks 1 & 2). In TRECs 3 and 4 the better automatic ad hoc runs involved “blind” feedback from the top few documents retrieved

Table 1: Automatic ad hoc: some training results, topics 151-200 on disks 1 and 2

Method	AveP	P5	P20	P30	P100	R-Prec	Rcl
"Long" topics							
Baseline: unexpanded topic term run	0.349	0.740	0.645	0.603	0.445	0.391	0.688
As above + passages	0.350	0.720	0.640	0.591	0.444	0.401	0.696
As row 1 but with pairs, plausibility threshold 10	0.348	0.744	0.647	0.597	0.443	0.392	0.690
As above but with passages	0.350	0.708	0.638	0.587	0.442	0.400	0.695
Expansion: 60 terms from 30 docs	0.408	0.760	0.686	0.647	0.483	0.429	0.744
As above + passages	0.415	0.760	0.687	0.651	0.482	0.432	0.752
As above but 50 terms	0.414	0.772	0.687	0.653	0.482	0.435	0.756
As above but 55 terms	0.416	0.768	0.690	0.649	0.485	0.437	0.755
TREC-3 cityal official result for comparison	0.401	0.740		0.625	0.476	0.422	0.739
"Short" topics							
Unexpanded topic term run	0.225	0.612	0.492	0.443	0.322	0.300	0.514
As above but with pairs, plausibility threshold 10	0.221	0.612	0.477	0.441	0.313	0.293	0.513
Expansion: 30 terms from 15 docs + passages	0.311	0.680	0.553	0.517	0.388	0.351	0.607

Table 2: Automatic ad hoc: some test results, topics 251-300 on disks 2 and 4

Method	AveP	P5	P20	P30	P100	R-Prec	Rcl
"Long" topics							
Unexpanded topic term run	0.231	0.508	0.360	0.322	0.206	0.273	0.473
As above but with passages	0.241	0.484	0.371	0.323	0.207	0.288	0.482
As row 1 but with pairs, plausibility threshold 10	0.233	0.504	0.366	0.326	0.210	0.277	0.481
As above but with passages	0.240	0.508	0.378	0.330	0.208	0.280	0.486
Expansion: 55 terms from 30 docs + passages (city96a1)	0.216	0.428	0.358	0.324	0.216	0.248	0.536
As city96a1 but terms from "best" passage of each doc	0.223	0.456	0.373	0.342	0.218	0.253	0.531
city96a1 evaluated on 47 topics	0.229	0.455	0.380	0.344	0.229	0.264	0.536
"Short" topics							
Unexpanded topic term run	0.152	0.368	0.283	0.248	0.157	0.193	0.383
As above but with passages	0.171	0.376	0.274	0.244	0.160	0.214	0.390
Expansion: 30 terms from 15 docs + passages (city96a2)	0.175	0.336	0.284	0.260	0.178	0.213	0.469
city96a2 evaluated on 47 topics	0.185	0.357	0.301	0.276	0.190	0.227	0.469

Table 3: Very large collection track: official results on test database

Method	P20
city96vlc1: 55 terms from 30 expansion docs, disks 2 & 4	0.262
city96vlc2: as city96vlc1 but disks 1-4	0.467

by a topic-term pilot search. In TREC-4 we tried manual "tweaking" of the queries thus obtained; this was not a success. Casting around for something else to try (other than procedures which involve non-statistical features of terms), it was decided to have another go at adding adjacent topic term pairs to the queries. We tried this in TREC-2 with neutral results, but this time pairs were restricted to those which occurred (in the databases) with frequency much higher than the expected value and a new pair-weighting formula was tried. However, results were still no better than those obtained using single terms only. In desperation, two long shots were tried: a form of term selection analogous to that used in the automatic routing, but without relevance information; and merging results obtained by different methods. Neither of these gave any improvement. However, repeats of TREC-3 runs gave results which were slightly better than our official TREC-3 run.

Training and official results are given in Tables 1 and 2 respectively, and City's position relative to the median scores in Table 6. VLC results are in Table 3.

4.2 Adjacent pairs

If s and t are two terms with frequencies $n(s)$ and $n(t)$ respectively the *plausibility* of the adjacent pair st is $n(st) \times C / (n(s)n(t))$, where C is the total number of tokens in the collection.

Example

The Narrative field of topic 2:

To be relevant, a document must discuss a currently proposed acquisition (which may or may not be identified by type, e.g., merger, buyout, leveraged buyout, hostile takeover, friendly acquisition). The suitor and target must be identified by name; the nationality of one of the companies must be identified as U.S. and the nationality of the other company must be identified as NOT U.S.

gives, at plausibility threshold 10, “friendly acquisition” with plausibility 44, “hostile takeover” with 939 and “leveraged buyout” with 4472 (and of course single terms).

4.2.1 Weights for term pairs

If a query contains a term pair as well as the individual terms, and all three are given “natural” (e.g. inverse collection frequency) weights, some allowance ought to be made for the mutual dependency between each term and the pair. In TREC-2 we tried several ways of adjusting pair or term weights [6], but these had little effect on the results. Steve Robertson suggested that a plausible pair weight would be $\log(n_{sANDt}/n_{sADJt})$ (where the n s this time are the numbers of posted documents), and this appeared to work a little better than giving both pairs and single terms their $w^{(1)}$ weights. However, the results were about the same as those from giving pairs just a “small” bonus weight.

4.3 Other ad hoc query generation methods

Query expansion

The top R documents from the pilot search were output and all terms other than stop and semi-stop terms extracted. These were $w^{(1)}$ -weighted in accordance with their occurrence in the top documents, with terms occurring more than once in the topic statement loaded with a k_3 value (usually either 8 or 1000 — varying within this range made little difference). The terms were then arranged in descending RSV value, and the required number taken from the top of the list, any term with RSV less than 3 being discarded.

Term selection after expansion

Some selection runs were done as follows. The top documents (typically 50 of them) retrieved by an assumed good set of queries from an expansion run, were treated as relevant, and a routing-type selection procedure done on terms from one of the expanded term sets. Scoring was usually on the top 100 documents retrieved at each stage in the selection procedure. Note that only one database was used instead of at least two as in the routing. The best runs were about as good as the best runs obtained by other methods.

Iterated expansion

A few runs were done using a second iteration of the expansion procedure. They were better than expected but not quite as good as the best singly-expanded runs.

Merging runs or queries

Several pairs of the better runs were merged, with results which were not very encouraging. However, the union of the best selection run and the best plain expansion run did give marginally the best results on most measures.

4.4 Very large collection track

Two runs were submitted: the baseline run on disks 2 and 4; and a “full” run on disks 1, 2, 3 and 4. The city96a1 run was re-used for the baseline run city96vlc1 (see Section 4.5), and the same method against the four-disk database for city96vlc2.

4.5 Ad hoc results

The new topics consist, like the TREC-3 ones, of title, narrative and description only. One of the runs submitted had to be based on the description field only.

City96a1 (and city96vlc1 and 2), using all topic fields, was based on 30 feedback documents, with a bonus to all topic terms (all that were in the expanded termset), any terms which did not occur in at least 3 of the 30 feedback

documents were rejected. Each query consisted of the top 55 terms, and passage searching was used. City96a2 ((desc) only) used the same method as city96a1 but with 15 feedback documents and 30 terms per query.

Trial results (Table 1), done on the TREC-3 ad hoc topics and test database, showed a small improvement over our best TREC-3 result. Query expansion was markedly beneficial, passage searching slightly beneficial and the use of topic term pairs neutral. The methods which gave the best trial results were used on the test database and topics for the official runs (Table 2).

The official results showed that query expansion gave higher recall but poorer low precision, while the effect of passages and pairs was much as in the trials. Expansion appeared to work a little better when terms were taken only from the best passage of each expansion document.

The VLC runs (Table 3) used the same method as city96a1. Note that the precision at 20 documents for the city96vlc1 run is not the same as that for city96a1 although the queries were the same. There appear to be two reasons for this:

1. the VLC assessments covered only documents retrieved in participants' "full" VLC run: thus, some documents retrieved in "baseline" runs were not assessed;
2. there are nearly 300 disagreements between the VLC relevance judgments and the official TREC-5 disks 2 and 4 qrels.

5 Automatic routing

5.1 Training database

The training database was constructed from disks 1-3 together with the TREC-4 routing set (without its Ziff, which is the same as disk 3's). This contains about $1\frac{1}{4}$ million documents. It was systematically partitioned into sub-databases in two ways: into halves, by selecting odd or even record numbers; and into thirds analogously.

5.2 Outline of procedure

Some control runs were done first, using topic terms and no relevance information (including some runs incorporating pairs of adjacent terms), and using topic terms reweighted in the light of relevance information.

The general procedure for finding a good query term selection method was similar to that of TREC-4, except that the partition of the training database into three parts was used.

term pool: a pool of terms was extracted from the known relevant documents in one third of the training database (the topic statements were not used²);

term selection: queries were generated by selection from the term pool using the second partition of the database; a number of different selection methods were tried;

evaluation: the queries resulting from each selection method were evaluated predictively against the third partition.

Having found a good selection method using the partition of the database into thirds, the partition into halves may be used to produce a final set of queries, thus making use of more of the available relevance information. Relevant documents from one half were used to produce the term pool and term selection was done on the other half. A set of queries generated by this method was submitted as City96r2. The filter runs and city96r1 consisted of queries produced using the partition into thirds.

Some training results are given in Table 4, official results in Table 5, with comparative results in Table 6.

5.3 Term selection measure and algorithms

In TREC-4 a number of different scoring measures were tried. None performed better than the TREC average precision, and this was the only measure used for TREC-5. However, a number of new selection algorithms were tried.

For all selection algorithms, the available terms were first arranged in descending RSV^3 [9] order to form a term pool of a given size. An initial query (sometimes empty) was formed from terms at the top of the pool, then terms

²During term selection a bonus was generally given to topic terms by using a non-zero value of k_3 in equation 3

³Note that RSV stands for Robertson Selection Value not Retrieval Status Value.

Table 4: Automatic routing: predictive results on part of training database

Method	AveP	P5	P30	P100	R-Prec	Rcl
Topic term runs						
Topic single terms, $k_3 = 1000$	0.309	0.616	0.507	0.387	0.365	0.717
Topic, single terms + pairs with Robertson pair wts (see 4.2.1), $k_3 = 1000$	0.315	0.616	0.511	0.391	0.370	0.723
Topic single terms $F4$ -reweighted, $k_3 = 0$	0.345	0.688	0.563	0.426	0.392	0.756
Runs with term selection						
Best fixed-weight run, pool 298, $k_3 = 3$	0.493	0.852	0.731	0.535	0.503	0.842
All weights varied on each iteration + 10 final passes, pool 100, $k_3 = 0$	0.496	0.852	0.724	0.535	0.508	0.848
Weights varied every 20-change, term pool 298, $k_3 = 3$	0.497	0.848	0.735	0.538	0.507	0.849
The above run merged with its inverse	0.504	0.852	0.737	0.543	0.507	0.854
3 runs and their inverses all merged	0.529	0.884	0.745	0.562	0.521	0.869
(The official city96r2 used the above method using half-databases)						
Individually "best" query for each topic (individualized k_1 and b)	0.547	0.924	0.768	0.567	0.537	0.872
(The official city96r1 used the above method)						

Table 5: Automatic routing: official results on test database

Method	AveP	P5	P30	P100	R-Prec	Rcl
city96r2 on 39 topics	0.386	0.661	0.532	0.380	0.415	0.736
city96r1 on 39 topics	0.372	0.692	0.508	0.371	0.405	0.719
city96r2 on 45 topics	0.347	0.582	0.465	0.331	0.360	0.736
city96r1 on 45 topics	0.329	0.600	0.444	0.322	0.351	0.719

Table 6: Automatic routing and ad hoc: comparisons with other participants' results

Routing (45 topics)												
	Precision at 100			Precision at 1000			Average precision			Precision at R docs		
	\geq median	=best	=worst	\geq med	=best	=worst	\geq med	=best	=worst	\geq med	=best	=worst
city96r1	43	5	0	44	12	1	41	3	1	45	9	0
city96r2	45	6	0	45	17	0	43	3	0	45	10	0
Ad hoc (50 topics)												
The medians for city96a1 are from the "long" topic runs, those for city96a2 from the "short".												
	Precision at 100			Precision at 1000			Average precision					
	\geq median	=best	=worst	\geq med	=best	=worst	\geq med	=best	=worst			
city96a1	41	12	0	39	18	1	35	8	0			
city96a2	39	4	1	42	10	1	34	4	1			

were added or removed in accordance with the algorithm being used. After each addition or removal a score was calculated (standard TREC average precision at cutoff 1000 on the set of known relevant documents for whichever sub-database was being used for selection).

In TREC-4, three types of iterative selection procedure were tried. These were, briefly

Find best. In each iteration every undropped term was examined. At the end of the iteration the highest scoring term was added to or removed from the query;

Choose first positive. In each iteration the first term which increased the score by an amount greater than a predetermined threshold percentage was added to or removed from the query;

Choose all positive (CAP). Every term was considered and immediately added to or removed from the query if it increased the score by more than the threshold. An iteration ended when all “live” terms had been considered.

There were a number of stopping rules, but, in practice, after a smallish number of iterations there would be no single term whose addition or removal increased the score.

Surprisingly, the last method (CAP) appeared from the TREC-4 work to be the best, and this has been borne out in TREC-5. So CAP was used for all runs after some preliminary trials. The new features in term selection were

- varying term weights during selection
- merging queries produced by different selection methods and database combinations.

Varying term weights during selection

Even with our increased computing power since TREC-4, anything at all comprehensive was out of the question. Various combinations of the following were tried (all during CAP selection):

- the weight of the term under consideration could be decreased or increased by given proportions; in some cases these proportions were varied (uniformly over all terms) during a selection run; typical proportions were $w \rightarrow .67w$ and $w \rightarrow 1.5w$
- a term not already included was tried at its original weight, then the lower weight and finally the higher weight; the term’s weight was set to the first one (if any) which gave a score increase
- every non-included term was tried at up to three weights
- all included terms were retried at a lower then a higher weight every time the current query had changed by a certain amount
- after the selection procedure had stopped a number of weight variation passes were done, with the variation proportions changed after any pass which gave increase in score.

Watching the progress of a selection run it often appeared that weight variation was going to be highly beneficial. However, term selection/rejection and weight variation tended to cancel one another out, and the net benefit from weight variation appears to be small. It seems that it is better not to be too radical: the best results are probably obtained by varying weights after the termset has changed by 20 or so, together with some passes at the end.

Merging runs

Merging a number of runs, usually in pairs – a run with its inverse – made a very significant improvement in the predictive results. The best single query set came from merging three pairs of runs. It seems likely that this is due to a reduction in the over-fitting effect which must result from such extensive use of the relevance information. When there are a large number of known relevant documents it may be advantageous to partition the training database further, using more than one partition during the term selection process.

5.4 Filtering

The city96r1 queries were run against the “predictive” third of the training database (the portion of the database not used as term source or for term selection). Each of the three utility functions ($R - 3N$, $R - N$ and $3R - N$, where R is the number of relevant documents retrieved and N the number of nonrelevant) was evaluated after each document was retrieved, and the lowest weight which corresponded to a maximum of each of the functions was submitted as threshold. This simple procedure produced fairly satisfactory results.

5.5 Results

Table 4 summarizes the results of a few of the training runs. It appears that weight variation as we tried it does not give much improvement over straight selection/rejection of terms. Runs using a considerable amount of weight variation on relatively small term pools (e.g. row 5 of the table) seem to give as good results as straight selection on large term pools (and give shorter queries).

The break-through (if that is the right way to describe it) came when we tried merging queries with their inverses. Although different and stochastically independent databases were used as source of terms and for selection, it is probably the case that the selected termset is closely “fitted” to the database used for selection. Merging queries from different selection databases must compensate for this effect. More investigation is needed. It may be that results could be improved by partitioning the training database still further, provided there are enough known relevant documents.

The official results (Table 5) look poor compared with previous years. However, the test database was very small, rather uniform and of limited scope. There were no relevant documents for at least four of the topics, and very few for six others. (Another topic missed assessment.) NIST sensibly evaluated results both on the 45 “non-empty” topics and on the 39 with at least eight relevant documents. In comparison with median scores from all full participants, city96r2 was as good as or higher than all medians on 43 of 45 topics and lower on average precision only on 2 of the low-posted topics; city96r1 slightly less good (Table 6).

6 Chinese track experiment

In this section, we describe the experiments with our Chinese text retrieval system modelled on Okapi. The experiments focus on applying the probabilistic model to Chinese text retrieval and comparing between two document-indexing approaches: word and single character approaches. About 175 megabytes of Chinese text and 28 Chinese topics were made available for runs in the ad hoc environment. Two City Chinese runs are submitted for evaluation, which are city96c1 for the word approach and city96c2 for the character approach. For evaluating the BM25 function, two new runs city96c12.BM25 and city96c24.BM25 are also generated for word and character approach respectively.

6.1 Automatic indexing

There are two kinds of methods for indexing Chinese text. One indexing method is to use words and phrases in texts as indexing terms to represent the texts. We refer to this method as the word approach. Since the Chinese words in a text are not separated by spaces, text segmentation, which divides text into linguistic units (commonly words and phrases), is regarded as a necessary precursor of word approach system [10]. To segment the Chinese text, a dictionary with about 70,000 Chinese words and phrases was built and a longest match algorithm for text segmentation was implemented. This algorithm is to extract a string of a certain number k (usually $k = 7$) of characters and to search for it in the dictionary. If it exists in the dictionary, the process continues with the next k -character text string. If it doesn't, the last character in the string is removed and the new string is then searched for in the dictionary. Figure 6.1 shows the longest match algorithm. According to the word segmentation produced by this algorithm, the Chinese TREC texts consist of approximately 43.6 million words and phrases.

The other method for indexing texts is based on single characters, in which texts are indexed by the characters appearing in the texts [11]. Usually each Chinese character is given a unique two-byte machine code. Therefore, it is a purely mechanical procedure to segment TREC Chinese text into single characters.

6.2 Probabilistic weighting functions

Basic weighting function

The basic weighting functions are based on the Robertson-Sparck Jones weight [5], which approximates to inverse collection frequency (*ICF*) shown in equation 4 when there is no relevance information.

$$ICF = \log \frac{N - n + 0.5}{n + 0.5} \quad (4)$$

The function above can be extended to include document length and within-document and within-query term frequency as providing further evidence, which have been shown to be highly beneficial in English text retrieval [12].

Figure 1: The longest match algorithm.

ALGORITHM: The Longest Match

INPUT: A string of Chinese characters $a_1a_2 \cdots a_k$;

OUTPUT: A set of Chinese words and phrases

begin

let i be 1 and j be k

while ($i \leq k$)

begin

if a_i is a digital character or English character

then call special_string_processing($a_i \cdots a_j$);

else if there is no character a_i in the index file of word segmentation dictionary or i equals to j

then put a_i into segmentation result file;

 increase i by 1;

else if there is a string $a_i \cdots a_j$ in the lexicon file of word segmentation dictionary

then put $a_i \cdots a_j$ into segmentation result file;

 let i be j ;

else decrease j by 1;

endwhile ;

end

One of these extended functions that has been implemented in our Chinese text retrieval system is:

$$w = \frac{tf}{(k_1 \cdot dl / avdl + tf)} \times \log \frac{N - n + 0.5}{n + 0.5} \times \frac{qtf}{(k_3 + qtf)} \oplus k_2 * nq * \frac{(avdl - dl)}{(avdl + dl)} \quad (5)$$

where

N is the number of indexed documents in the collection,

n is the number of documents containing a specific term,

tf is within-document term frequency,

qtf is within-query term frequency,

dl is the length of the document,

$avdl$ is the average document length,

nq is the number of query terms,

the k_i are tuning constants, empirically determined, and

the \oplus in the formula indicates that the following component is added only once per document, rather than for each term.

This formula now defines the weight of a term (that is, the contribution of that term to the total score for the document) in the context of an individual document. This is the formula referred to as BM11 in our TREC-3 work [7]. Some results are shown below with the later BM25 function

Phrase weighting

Okapi's usual method for dealing with phrases in English has been as follows. If a phrase is identified at indexing time, it is indexed as a single unit, and the component words of the phrase are not normally indexed. If it is identified only at search time, it may be searched as a single unit by means of an adjacency operator. Again the words may not be normally searched on their own, although some variations have been tried.

It would be reasonable to seek a method that allowed a match on a phrase or on either or all of its component parts. It might then be reasonable to assume that, for a phrase consisting of two terms t_1t_2 ,

$$w(t_1), w(t_2) \leq w(t_1 \wedge t_2) = w(t_1) + w(t_2) \leq w(t_1 \text{ adj } t_2).$$

Here *adj* is the adjacency operator. The equation in the middle represents the usual scoring method for the \wedge (and) operator: the score assigned to a document is the sum of the weights of the matching terms. The assumption is that the phrase carries a higher score than the two terms separately.

The problem of devising such a method consistent with the probabilistic model has not generally been tackled in text retrieval in English. But for text retrieval in Chinese, the problem is likely to be more serious than it is in English. This would be so in a word-based system, since there are likely to be considerable differences between Chinese speakers as to whether a given combination of characters is actually a single word or a phrase. But it is even more serious in a character-based system, where one would want a match on a complete word or phrase to carry a higher score than matches on the component characters.

Weighting functions for Chinese TREC experiment

We need, therefore, a weighting function which will enable us to cope with phrases in a word-based system, and with words or phrases in a character-based system. Suppose, then, that we have a sequence of j adjacent units $t_1 t_2 \dots t_j$ (characters or words) constituting a single larger unit (word or phrase). In the Robertson/Sparck Jones model, each unit (large or small) has a "natural" weight, given by the formula; let these be w_{t_i} and $w_{t_1 t_2 \dots t_j}$ respectively. Then we can suggest a number of weighting functions which satisfy (or will probably satisfy) the above condition or something like it. Table 7 gives a few such functions which have been implemented.

Table 7: Weight methods

Weight methods	$w(t_1 \text{ adj } t_2 \text{ adj } \dots \text{ adj } t_j)$	$w(t_1 \wedge t_2 \wedge \dots \wedge t_j)$
<i>Weight</i> ₀	$\sum_{i=1}^j w_{t_i} + j^k * w_{t_1 t_2 \dots t_j}$	$\sum_{i=1}^j w_{t_i}$
<i>Weight</i> ₁	$w_{t_1 t_2 \dots t_j}$	$\sum_{i=1}^j w_{t_i} - j^k$
<i>Weight</i> ₂	$w_{t_1 t_2 \dots t_j}$	$\sum_{i=1}^j w_{t_i}$
<i>Weight</i> ₃	$\sum_{i=1}^j w_{t_i} + \log \frac{\#(t_1 \wedge t_2 \wedge \dots \wedge t_j)}{\#(t_1 \text{ adj } t_2 \text{ adj } \dots \text{ adj } t_j)}$	$\sum_{i=1}^j w_{t_i}$
where $\#(t)$ indicates the number of documents containing the term t and k is another tuning constant		

None of these functions has a very strong probabilistic basis, beyond the attempt to satisfy the condition. Each has two versions, one applied to words and the other to characters, so there are eight functions in all. The "natural" weights come from equation 5: w_{t_i} is obtained by applying the equation to term t_i , and $w_{t_1 t_2 \dots t_j}$ comes from applying the same equation to the combined term $t_1 t_2 \dots t_j$.

In the Chinese TREC experiments, we chose two retrieval results which use *Weight*₀ as the weighting method for evaluation. Further experiments suggested that *Weight*₁ might be preferable, and the BM25 results reported below also use *Weight*₁.

Query processing

There are two kinds of automatic methods for query processing in our Chinese text retrieval system. One is character-based method, which uses character, character pairs and multi-character adjacencies as retrieval keywords. The other is word-based method. Given a word segmentation system, similar methods for characters can be applied to words as a way of allowing phrases to contribute to the matching. We refer to this method as the word-based query processing. Table 8 shows nine methods which have been implemented in our systems.

We use M_5 and M_6 which are word-based in our Chinese TREC experiments. The text in all parts of the Chinese topics were treated in the same way. Text segmentation is applied to segment the topics into the words and phrases. Only pairs of the adjacent segmented terms which occur in the same subfield of the topic are regarded as new potential phrases. All these segmented terms and new potential phrases are ranked by the values of their weights multiplied by the within-query frequencies. The top 36 terms are used as keywords for searching the word index and for searching the character index using the adjacency operator.

Table 8: Illustration of retrieval algorithms

Algorithms	Query processing	Database	Number of weighting methods
M_0	characters	character	1
M_1	characters+character-pairs	character	8
M_2	characters+multi-character adjacencies	character	8
M_3	words	character	8
M_4	words	word	1
M_5	words+word-pairs	character	8
M_6	words+word-pairs	word	4
M_7	words+multi-word adjacencies	character	8
M_8	words+multi-word adjacencies	word	4

6.3 Experimental results

The two submitted Chinese results are shown in Table 9 and Table 10. Both runs are around the median score of the groups participating in the Chinese task. The results for the two runs based on the BM25 weighting function are shown in Table 10. By using a combination of the BM25 function and $Weight_1$ for phrases, the word and character approach perform 18% and 21% better than the word approach city96c1 respectively. Taking the word approach city96c1 as baseline, the character method city96c2 shows a slight improvement. However, the change to BM25 (essentially the equation given as 1) produces a much greater improvement.

Table 9: Comparative Chinese results

Run	Best	> median	= median	< median
<i>city96c1</i>	4	9	3	12
<i>city96c2</i>	0	11	7	10

Table 10: Chinese ad hoc results

Run	Average Precision	Total Rel Retrieved	R Precision	Precision 100 docs
<i>city96c1</i>	0.3256 (+0.0%)	1891	0.3418	0.2900
<i>city96c2</i>	0.3336 (+2.5%)	1950	0.3493	0.2939
<i>city96c12.BM25</i>	0.3828 (+17.6%)	1997	0.3893	0.3175
<i>city96c24.BM25</i>	0.3950 (+21.3%)	2015	0.4053	0.3250

6.4 Conclusions

The application of probabilistic retrieval methods in free-text Chinese language material is very successful. In terms of average precision and total number of relevant retrieved documents, the character approach does better than the word approach. As the results indicate in Table 10, the combination of BM25 and $Weight_1$ for phrases makes a significant positive contribution to the quality of retrieval.

7 Interactive track experiment

The task definition and the type of users conducting the searches were the main differences for this round of Okapi interactive TREC. The set task for the TREC-5 interactive track was defined as finding relevant documents which covered as many different aspects of a topic as possible within a twenty minute search session. The eight searchers were postgraduate students in the Information Science Department. They had some experience of information retrieval systems, but acted as end-users. Each of the test topics was carried out by two searchers. Past performance in interactive searching had shown that the query expansion facility in the Okapi system was effective in retrieving more similar items and it was envisaged that the task of finding different aspects would be more taxing. The same GUI interface was used as in TREC-4 with some modifications to allow for greater user control in the search interaction as indicated below:

Incremental query expansion In previous versions of the interactive interface query expansion had to be explicitly requested. Clicking on an EXPAND button caused the current query to be merged with all terms extracted from new relevant documents. In incremental query expansion, introduced in the current interface for TREC-5, once the searcher makes a positive relevance judgment, extracted terms are automatically added to the working query and current query terms are dynamically re-weighted and re-ordered if appropriate (See Appendix A.2.6). The intention was to make the underlying relevance feedback process more visible by enabling the searcher to relate relevance judgments more closely to the working query.

Changing relevance judgments and restoring query terms In manipulating the working query, searchers were given greater flexibility to ‘undo’ previous decisions. Firstly, they could change previous relevance judgments and switch between “Full Document” or “Passage Only” for term extraction. Secondly, terms removed by the searcher from the current query were displayed in a separate window so that they could easily be reinstated in the query if required at a later stage.

As in the previous interactive round, searchers filled in a post-search questionnaire indicating their evaluation of each of the query tasks. The results of the questionnaire are found in Appendix D. The analysis in the following section is an attempt to classify the level of difficulty for each of the twelve queries according to the searchers’ perceptions and to compare and correlate queries deemed difficult or easy with searching behaviour characteristics.

7.1 Topic classification

In five of the post-search questions searchers were asked to rate different aspects of the search task as ‘easy’, ‘moderately easy’, ‘difficult’ or ‘very difficult’. These ratings related to the following questions (See Appendix D):

- interpreting the topic initially (Q1),
- generating initial query terms (Q2),
- finding relevant items from the initial hitlist (Q3),
- finding documents on different aspects of the topic (Q5), and
- rating the overall search task (Q9).

Points were allocated for each of the responses ranging from 0 for the ‘easy’ category to 3 for the ‘very difficult’ category. Table 11 shows the resulting score and ranking for each of the twelve test topics with the most difficult scoring 21 and the least difficult scoring 4. For the three most difficult and the three easiest topics there was a high degree of agreement between the two searchers who undertook the individual topics. However for the three most difficult topics, searchers were much less consistent in indicating how difficult it was to interpret in the first instance. Difficult and easy topics were evenly distributed amongst the different blocks allocated to the searchers.

Table 11: Difficulty rating and ranking of topics

topic	256	292	260	255	254	284	286	299	264	293	294	258
score	21	21	17	16	14	13	11	11	10	9	7	4

7.2 Searching behaviour for 'difficult' and 'easy' topics

An analysis of the three highest and three lowest ranking topics was undertaken to compare any associated differences in searching behaviour. The characteristics considered included: the generation of initial query terms, the use of query expansion, the manipulation of the query term sets, and the examination of retrieved documents

7.2.1 Generation of initial query terms

Table 12 compares the source of initial query terms for the 'difficult' and 'easy' topics undertaken by the pairs of searchers, as well as the overall figures for all twelve topics. It appears that for 'difficult' topics more query terms were generated initially than for the 'easy' topics but that a smaller proportion of query terms were derived from the topic specification as given. Moreover searchers used a greater proportion of phrasal terms to generate 'easy' queries than 'difficult' ones. As in TREC-4, problems arise in expressing abstract ideas and the length of the topic specification itself seems to have a direct bearing on the number of terms used for the initial query formulation. The mean for TREC-5 was 7.06 compared to 4.72 for the shorter TREC-4 topics.

Table 12: Source of terms for the initial query

Topic category	No. of terms	Topic spec.	User generated	No. of phrases
Difficult	Mean	9.5	6	3.5
	Median	10	5.5	3.5
	Range	4—13	3—9	0—8
	Total	57	36 (63%)	21 (58%)
Easy	Mean	5.8	4.5	1.3
	Median	6	5	1
	Range	1—10	1—7	0—3
	Total	35	27 (77%)	8 (23%)
All topics	Mean	6.9	5.2	1.8
	Median	6	5	1.5
	Range	1—19	1—15	0—8
	Total	166	124 (75%)	42 (25%)

7.2.2 Query expansion and manipulation of query term sets

The query expansion facility was used by all the searchers for all the topics except one. The number of iterations (i.e. new search based on an expanded query with system extracted terms) for each search session averaged 3.63. This is comparable to the interactive searches for TREC-4 (3.56) and shows that there is little difference between expert and end-users in their dependency on the system to support query reformulation. There was also little difference between the mean number of iterations undertaken for 'difficult' (3.8) and 'easy' (4.2) topics.

Overall query expansion was deemed by searchers to be helpful in finding relevant documents in two-thirds of the topics (Table 13). A breakdown of the two categories of topics shows little agreement between searchers, although query expansion may appear to be more helpful in the case of 'easy' topics.

Table 13: Perceived effectiveness of query expansion

Topic category	Helpful	Not helpful	No expansion
Top 3 difficult	3	3	0
Bottom 3 easy	4	2	0
All topics	15	8	1

Searchers could manipulate query term sets in three ways: by removing system generated candidate terms, by adding their own new terms or restoring previously deleted terms. In this current round of TREC users removed a substantially lower number of terms from the working query, an average of 25.22 compared to 39.32 in TREC-4. Although the top 20 terms of an expanded query were displayed in both cases, the incremental query expansion may

have produced less noise. However Table 14 shows that five time as many terms were removed from term sets for the 'easy' topics as opposed to the 'difficult' ones. This could perhaps indicate that for 'easy' topics searchers had a much better idea of what terms were appropriate for the task. Indeed the inappropriateness of terms is given as the most prominent reason for deleting query terms overall (Table 15). The least prevalent reason was because aspects of the topic had already been covered. Searchers of 'easy' topics also agreed that the automatic display of new query terms helped them identify new aspects of the topic whereas those undertaking 'difficult' topics were more divided.

Table 14: Removal of terms from working query

Topic category	Total	Mean	Median	Range
Difficult	55	9.2	3	1—41
Easy	275	46.0	29.5	7—109
All topics	166	6.9	6	1—109

Table 15: Reasons for term removal

Topic category	Proper names	Numbers	Inappropriate	Already covered	None removed
Difficult	1	2	5	1	0
Easy	4	3	5	2	0
All topics	11	11	20	3	1

By contrast user generated query terms were rarely deleted and the number of removed terms restored was also negligible. Searchers also concentrated on the working query as presented by the system and introduced few of their own query terms in the course of a search (See Appendix C.2).

7.2.3 Examination of retrieved documents

A hitlist of fifty documents was displayed at each iteration. On average searchers scrolled through a total of 41.2 titles in a search session of approximately twenty minutes. This is proportionately comparable to the average of 54.2 items scrolled in the thirty minute sessions in TREC-4. However the very different nature of the interactive retrieval task set in the two rounds is manifested by the number of full documents seen. In TREC-5 the mean was almost half of that in TREC-4, 14.13 documents as opposed to 31.12. Evidence that searchers were very selective in examining full documents for the TREC-5 interactive task is also provided by the fact that a number (19.5%) of search iterations led to hitlists from which no full documents were examined. With regard to 'difficult' and 'easy' topics there was little difference between the number of documents 'viewed' and 'seen', or how far searchers scrolled.

Searchers rejected 51% of 'seen' documents. Although for feedback purposes searchers were given the choice of assigning relevance to full documents or best passages, in 87% of instances the full document was chosen.

7.3 Results of interactive searches

The precision and aspectual recall for the interactive searches (excluding Topic 284) were 0.487 and 0.330 respectively. The average precision and aspectual recall for 'difficult' and 'easy' topics indicate a lower performance for 'difficult' topics, 31.3% compared to 43.1% for precision and 22.3% compared to 38.1% for recall.

Because of the different nature of the interactive task these results are not comparable with other TREC results. However one observation has emerged which requires further consideration. Disagreement between assessors' aspectual judgments and those of the searchers was found for 136 out of a total of 583 documents. There was also disagreement between the initial binary relevance assessments by the assessors and their subsequent assessment of aspects. Ten of the documents were judged to be relevant but to contain no aspects, whilst 126 documents were judged as not relevant but to contain aspects.

A comparison between the 'difficult' and 'easy' topics taking account of this inconsistency, reveals that there was a total of 47 disagreeing judgments for the former and 11 for the latter. It would seem that an inherent feature of a difficult topic or search is one where there are discrepancies on how the search task is perceived by the searchers as well as disagreements on relevance judgments. There was, however, no apparent association between the number of aspects identified and the difficulty of the topics.

References

- [1] Mitev N N, Venner G M and Walker S. *Designing an online public access catalogue: Okapi, a catalogue on a local area network*. British Library, 1985. (Library and Information Research Report 39.)
- [2] Walker S and Jones R M. *Improving subject retrieval in online catalogues: 1. Stemming, automatic spelling correction and cross-reference tables*. British Library, 1987. (British Library Research Paper 24.)
- [3] Walker S and De Vere R. *Improving subject retrieval in online catalogues: 2. Relevance feedback and query expansion*. British Library, 1990. (British Library Research Paper 72.) ISBN 0-7123-3219-7
- [4] Robertson S E *et al.* Okapi at TREC. In: [13] (pp.21–30).
- [5] Robertson S E and Sparck Jones K. Relevance weighting of search terms. *Journal of the American Society for Information Science* 27 May–June 1976 p129–146.
- [6] Robertson S E *et al.* Okapi at TREC–2. In: [14]. p21–34.
- [7] Robertson S E *et al.* Okapi at TREC–3. In: [15]. p109–126.
- [8] Robertson S E *et al.* Okapi at TREC–4. In: [16]. p73–96.
- [9] Robertson S E. On term selection for query expansion. *Journal of Documentation* 46 Dec 1990 p359–364.
- [10] Wu Z and Tseng G. Chinese text segmentation for text retrieval: Achievement and Problems. *Journal of the American Society For Information Science* 44 (9) p532–541 1993
- [11] Chen G. On single Chinese character retrieval system. *Journal of Information* 11 (1) p11–18 1992 (in Chinese).
- [12] Robertson S E, Walker S and Hancock-Beaulieu M. Large test collection experiments on an operational, interactive system: OKAPI at TREC. *Information Processing & Management* 31 (3) p345–360 1995.
- [13] *The First Text REtrieval Conference (TREC–1)*. Edited by D K Harman. Gaithersburg, MD: NIST, 1993.
- [14] *The Second Text REtrieval Conference (TREC–2)*. Edited by D K Harman. Gaithersburg, MD: NIST, 1994.
- [15] *Overview of the Third Text REtrieval Conference (TREC–3)*. Edited by D K Harman. Gaithersburg, MD: NIST, April 1995.
- [16] *The Fourth Text REtrieval Conference (TREC–4)*. Edited by D K Harman. Gaithersburg, MD: NIST, October 1996.

A Interactive System Description

A.1 GUI interface

The interface is an adaptation of the Trec 4 GUI to the BSS written in C, C++ and TCL/TK. Figures 2 and 3 show screen dumps of the running system. It is composed of six main windows some of which contain context-enabled ‘function’ buttons.

A.1.1 Query Entry

A.1.2 Working Query

A scrollable, ranked list of the terms that constitute the current working query.

A.1.3 Removed Terms

A scrollable list of any terms removed by the user from the working query, displayed in the order of removal.

A.1.4 Hitlist

A scrollable, ranked hitlist for the current iteration. Once documents have been seen from this list, the entry at the top of the window is the next ranked document after the last one seen.

A.1.5 Pool of Positive Relevance Judgments

A scrollable, ranked list of positive user relevance judgments.

A.1.6 Full Document

A scrollable window in which full documents selected from the hitlist are displayed. Items A.1.1 to A.1.5 are all displayed in one window together with two context-sensitive buttons to:

- **SEARCH:** Search on the current working query to form the hitlist for the next iteration. This button is enabled every time the working query is modified, either by the entry of new user terms, or the (automatic) addition/removal of system extracted terms.
- **EXIT:** Always enabled.

Window A.1.6 is a pop-up text window for the display of full records. At the bottom of the window are three buttons for making relevance judgments. These are described in more detail in Section A.2.4.

A.2 User interaction

A.2.1 User input of query terms

Users enter one or more terms, optionally followed by an adjacency operator (a '+' sign) as the last non-space character in the line, into the query entry box.

- No adjacency operator. Each term is treated as a single member of the query.
- An adjacency operator. The set of terms are treated as:

ADJ : a phrase, possibly including intervening stopwords. The terms are taken in input order.

SAMES : within the same sentence. The terms may occur in any order within the sentence.

For example, the input line "laws of the sea" would find documents that included, amongst others, both "laws of the sea" and "The right of innocent passage in Black Sea waters is not provided for by Soviet law". Any set of terms input with an adjacency operator is referred to as a 'PHRASE'. Internally two sets are generated, $S(A)$ and $S(S)$, with number of postings and weights $N(A)$, $W(A)$ and $N(S)$, $W(S)$ respectively. These are then combined into one set according to the rules:

Sets Generated	Use
1. $N(A) = N(S) = 0$	both discarded
2. $N(A) = 0, N(S) > 0$	$S(S), N(S), W(S)$
3. $N(A) = N(S), N(A) > 0$	$S(A), N(A), W(A)$
4. $N(A) < N(S), N(A) > 0$	$S(A), N(A), W(A), S(S)-S(A), N(S)-N(A), W(S)$ (Count as one term only.)

The weight calculated for each term is a Robertson/Sparck Jones F4 predictive weight, with halves (equation 2). In addition, user entered terms — both single terms and phrases — are given a loading; r and R increased by 4 and 5 respectively. Terms are displayed in the working query window in descending order of Robertson Selection Value [9]. Information displayed with each term is (a) the number of postings, and (b) the number of documents judged as relevant that contain the term. Phrases are followed by a letter indicating:

A Adjacency only

S Same sentence occurrence only

B Both adjacency and same sentence

G An indexed phrase.

A.2.2 Searching

The working query is composed of N terms ($N \leq 20$) from the entire termset made up of user-entered and extracted terms. Before each reformulation of the working query the termset is first ordered by two sort keys:

- **USER_TYPE** (User or Extracted) descending
- **RSV** descending

The rules for the inclusion of terms from the termset in the working query are as follows. Suppose the entire termset is composed of T terms made up of U user-entered terms and E extracted terms. i.e. $T = U + E$. There are two cases to consider:

$U < 20$: All of the user-entered terms are allowed into the working query. For extracted terms (if any) to take up the remaining $20 - U$ places they must satisfy the conditions that:

1. They occur in at least two relevant documents.
2. The RSV of the term is at least 60 percent of the average RSV of the current working query.

$U \geq 20$: The working query will be composed of the top 20 user-entered terms.

A search on the working query entails combining the terms in a best match operation (bm25). Passage retrieval (Section 3.1) is applied to the document set generated with parameters $p_unit = 4$, $p_step = 2$, $k_1 = 1.6$ and $b = 0.7$. The weight of the document is taken to be the higher of the weights of the full record or the best passage.

A.2.3 Hitlist generation

For a search performed on any iteration, documents are included in the hitlist only if they satisfy the two conditions:

1. they do not exist in a set generated by a previous iteration.
2. they satisfy a length threshold condition, i.e. the full record or, for longer documents, the best passage, must be less than 10K characters in length (in contrast to Trec 4 where all documents retrieved were included).

The hitlist is made up of the top H ranked documents ($H \leq 50$) that satisfies both of these two conditions. An entry for each document consists of:

A header line. <record_no> <docid> <normalised_weight> <document_length> [<passage_length>]

The <normalised_weight> is the system weight mapped onto the range 1..1000. The <document_length> and <passage_length> are given in pages, where one page is approximately 2000 characters.

A system generated title. Since generally records have no distinct title field, a "title" for display in the hitlist window is made up from approximately the first 150 characters from the headline field (if any) and the text field.

Query term occurrence information. This may (apparently) contain some duplicate information since the information is obtained by counting the highlighted terms in the document, which may contain different sources for the same stemmed terms.

A.2.4 "Seeing" documents

A document, selected for "seeing" by double-clicking anywhere in its hitlist entry, is displayed in a pop-up, scrollable text window. The best passage (or the whole document if the same) is highlighted in light-cyan; query terms are highlighted in green. The document is displayed either at the start line of the best passage, or, if (i) the best passage is the whole document or (ii) there is no best passage, the line containing the first query term. At the bottom of the text window are three buttons to allow users to make a relevance judgment.

1. **Full Document:** Relevant, terms are extracted from the text field of the entire document.
2. **Passage Only:** Relevant, terms are extracted only from the best passage.
3. **Not Relevant:** Not relevant.

Searchers must make a relevance judgment before they may go onto to any other part of the search process. A relevance judgment can be altered (e.g. $F \rightarrow P$, $P \rightarrow F$, $[F | P] \rightarrow N$, or $N \rightarrow [F | P]$) at any time until a new search is made, in which case the set of extracted terms is altered accordingly.

A.2.5 Relevance judgments pool

The normalised ranked hitlist information for all documents currently judged as relevant. Any member of the current relevance judgments pool that exists in a document set generated by a new search, has its weight set to that which it has in the latest document set; the display order is adjusted accordingly.

A.2.6 Incremental Query Expansion

Once the searcher has made two or more positive relevance judgments — "Full Document" or "Passage Only" — extracted terms are automatically added to the working query if they satisfy the conditions specified in A.2.2. After each relevance judgment all terms extracted from the appropriate section of the document are merged with the existing termset and the relevance information, including weights and RSVs, is adjusted accordingly.

A.2.7 Removing terms

Terms may be removed from the working query by double clicking on its entry in the query window. Removed terms are displayed in the removed terms window (A.1.3). It is possible that in the entire set of user-defined and extracted terms there are more than 20 candidate terms for the working query. Thus, as terms are removed, other terms may be promoted to take their place. A removed term may be reinstated in the query by double-clicking on its entry in this window.

A.2.8 Quitting

Quitting the search is achieved by clicking once on the "Exit" button.

B Experimental Conditions

B.1 Searcher characteristics

Eight searchers, all post-graduate students in the Department of Information Science, were divided into two groups — A and B. Group A consisted of one male and three females; Group B consisted of two males and two females. Their ages ranged from middle 20s to late 30s. The searchers were end-users who had (a) no specialist knowledge of any of the subject domains covered by the topics, and (b) were not familiar with the Okapi system of searching.

B.2 Task description/training

Within each group each searcher was randomly assigned one of the four blocks of three topics: i.e. there were two searchers allocated to each block. Group A carried out their training and searches during the week beginning 29th August, Group B during the week beginning 9th September. Each searcher was given training which consisted of:

1. a set of guidelines which included information such as:
 - (a) the task definition as defined for the interactive track,
 - (b) procedures to follow in the event of a system crash,
 - (c) suggestions on strategies which could be adopted for carrying out searches, e.g. to remove terms deemed as irrelevant from extracted lists of terms before carrying out an expanded search.
2. the opportunity to familiarize themselves with the interface and the task by carrying out three trial runs on topics taken from blocks that they would not be searching themselves.

Three searchers carried out their training and performed their runs on the same day. The other five searches performed their runs on the day after their training. After each "real" search, each searcher was asked to fill in an evaluation questionnaire for the search before proceeding on to the next one.

C Search Process

Unless otherwise stated the column 'Type' entries N, + and A refer to:

N: terms defined with no adjacency operator

+: terms defined with an adjacency operator

A: all terms defined.

C.1 Clock Time

Mean	Median	Variance	Range
20.6	20.8	3.1	13.9-22.9

Times are given to the nearest tenth of a minute.

C.2 Number of User Defined Terms

Type	At all iterations				After the first iteration			
	Mean	Median	Variance	Range	Mean	Median	Variance	Range
N	4.96	3.5	21.78	0-16	2.38	1	10.77	0-14
+	3.83	3	9.80	0-13	1.21	1	3.74	0-9
A	8.79	8	24.26	2-18	3.59	2.5	15.04	0-15

"Phrases" Defined By Searchers

Phrases generated: 94 Phrases used: 74

C.3 Number of Terms Used In Queries

Type	Initial query				Final query			
	Mean	Median	Variance	Range	Mean	Median	Variance	Range
N	4.56	2.5	6.96	0-13	15.13	17	21.33	4-20
+	2.50	1.5	5.50	0-13	3.08	2	7.56	0-13
A	7.06	5.5	20.30	1-19	18.21	20	11.48	8-20

C.4 Number of Iterations

An iteration, i.e. a new query formulation, was taken to be marked by each 'search' command. Expansion was performed incrementally. No data was kept to indicate how many times and when each working query was altered by the inclusion/exclusion of extracted terms.

Mean	Median	Variance	Range
3.63	3	1.98	1-7

C.5 Documents "Viewed" (hitlist) and "Seen" (full text)

"Viewing" a document consisted of seeing a header line, a system generated title, and query term occurrence information as described in Appendix A.2.3. The figures represent the percentage distance scrolled through the hitlist by the searcher. "Seeing" a document consisted of showing the full record in a scrollable window.

Viewed				Seen			
Mean	Median	Variance	Range	Mean	Median	Variance	Range
60.58	67	835.99	8-98	14.13	13.5	15.42	9-21

C.6 Relevance Judgments

In the following table the relevance judgments column ('Rel') are given as: **F** (Full document), **P** (Passage) and **N** (Not relevant).

Rel	Mean	Median	Variance	Range
F	5.96	6	9.69	0-14
P	0.92	0.5	2.17	0- 6
N	7.25	7	10.46	2-16
All	14.13	13.5	15.42	9-21

C.7 Use of System Features

Command	Mean	Median	Variance	Range
define - N	4.96	3.5	21.58	0-16
+	3.83	3	9.80	0-13
A	8.79	8	24.96	2-18
search	3.63	3	1.98	1-7
show	14.13	13.5	15.42	9-21
remove	25.22	13.5	1119.18	0-116
restore	0.85	0	3.40	0-8

C.8 Number of User Errors

Data on user errors were not collected. However, there was one search during which the system "crashed". This occurred almost at the end of the 20 minute period and so this was taken as the end of the search.

C.9 Topic 274

C.9.1 Search Narrative

The initial query terms entered by the searcher for this topic, concerned with latest developments in the production of electric automobiles, were 'electric', 'automobiles', 'batteries', 'production', 'feasibility' and 'recharging', all of which were chosen from the topic specification itself. The searcher also included a term and phrase of her own, 'cars' and 'alternative power'. Only one document was viewed and judged relevant from the first hitlist. The term 'problems' was added for the second iteration, which produced a fruitful hitlist. Eight out of the top ten ranking documents were examined and seven were deemed relevant. The rejected document on battery technology had been judged as relevant initially and was reconsidered in the light of other documents concerned with batteries.

As each positive relevance judgment could affect the weighted list of extracted candidate query terms, in the second iteration the searcher deleted a total of 107 terms from the term set on seven occasions in the course of finding 10 relevant documents. These included proper nouns, dates as well as other terms which were deemed not be appropriate for the topic. The rationale given by the searcher was "to avoid narrowing down the query or taking it into a different direction". For example, 'prototype', 'Los Angeles' and 'Chrysler' were considered to be too narrow whereas 'driving', 'characteristics' and 'highway' were judged as not directly relevant.

Following the retrieval of seven relevant items from the top ranks, it became more difficult to find others which dealt with different aspects of the topic and the searcher had to scroll as far as the 38th item in the hitlist to find three more relevant documents. More terms were also removed from the term set at this later stage. The searcher indicated that the replacement of removed terms by others lower down the ranking was annoying and she was prompted to delete them as a matter of course. However when the term 'environmental' came up the searcher commented, "I was surprised that such a relevant term didn't come up earlier". Apparently she had not considered introducing it herself. The ranking of system generated terms above user generated ones caused some disquiet as well as the fact that some relevant terms were removed by the system following a subsequent positive relevance judgment.

In the third iteration seven more full documents were seen and three more were judged relevant. Only two more terms were removed from the term set. Some of the documents rejected were relevant but did not add new information for the topic. In all cases in making positive relevance judgments, the searcher chose to mark the full document as relevant for relevance feedback purposes as opposed to the highlighted best passage only. In viewing the printed output of the search the searcher admitted that due to time constraints, she had judged relevance in the latter stages of the test on the strength of terms present and their distribution throughout the document.

The search started off very successfully and was generally considered to be easy. The display of extracted query terms helped to identify new aspects of the topic and query expansion led to finding more documents and improve the coverage of the topic.

C.9.2 Breakdown of command usage

Define (N) 2 Define (+) 1 Search 3 Show 17 Remove 109 Restore 0

D Search Evaluation: questionnaire results

1. How difficult was it to interpret initially as given?

Easy	Moderately easy	Difficult	Very difficult	TOTAL
9 (37%)	10 (42%)	2 (8%)	3 (13%)	24 (100%)

2. How difficult was it to generate initial search terms?

Easy	Moderately easy	Difficult	Very difficult	TOTAL
9 (37%)	9 (37%)	5 (21%)	1 (4%)	24 (100%)

3. How difficult was it to find relevant items from the initial hitlist?

Easy	Moderately easy	Difficult	Very difficult	TOTAL
3 (12.5%)	12 (50%)	6 (25%)	3 (12.5%)	24 (100%)

4. How did you identify the different aspects of the topic?

From the initial specification	21	88%
From the initial hitlist	2	8%
By examining documents	10	42%

5. How difficult was it to find documents on the different aspects of the topic?

Easy	Moderately easy	Difficult	Very difficult	TOTAL
3 (12.5%)	9 (37.5%)	7 (29%)	5 (21%)	24 (100%)

6. If you removed terms from the extracted term lists on what basis did you do so?

Proper names	Numbers	Difficult Inappropriate	Aspects already covered
11 (46%)	11 (46%)	20 (83%)	3 (12.5%)

7. Did the automatic display of new query terms help identify new aspects of the topic?

Yes	No	TOTAL
13 (54%)	11 (46%)	24 (100%)

8. Did the use of expanded searching lead to finding documents covering new aspects of the topic?

No expansion	Expansion helpful	Expansion unhelpful	TOTAL
1 (4%)	15 (63%)	8 (33%)	24 (100%)

9. How would you rate the overall difficulty of the topic question?

Easy	Moderately easy	Difficult	Very difficult	TOTAL
1 (4%)	12 (50%)	7 (29%)	4 (17%)	24 (100%)

10. How would you rate the success of your search?

Successful	Moderately successful	Not successful	TOTAL
5 (21%)	13 (54%)	6 (25%)	24 (100%)

Elapsed time — 006:28	
Type one or more words, or a single phrase (followed by a + sign) to add to your query, then press return	
<p>Query Terms</p> <p>533 3 : recharging 3246 3 : batteries 444 2 : cadmium 564 2 : alternative power (B) 1618 2 : nickel 18083 3 : vehicle 2214 2 : acid 22197 3 : utilized 25845 3 : electric 26090 3 : alternative 26285 3 : cars 39721 3 : practical 43726 3 : environmental 44062 3 : Technology 5656 2 : emissions 5722 2 : fleets 111372 3 : production 4211 1 : automobiles 71350 2 : problems 5308 0 : feasibility</p>	<p>Document Hitlist</p> <p>WSJ910826-0083 912 1 page Nissan Unveils Electric Car Claims 'Fastest' Recharge DETROIT (AP) — Nissan Motor Co. unveiled a car powered by an electric battery that can be recharged in 15 minutes — a..... Electric (7) Car (7) Recharge (3) battery (2) recharged (2) cars (1) problems (1) batteries (1) production (1)</p> <p>FT 14 DEC 94 / Business and the Environment: Drive to overcome</p> <p>1998 Chrysler GM</p> <p>[F] 1000 FR940922-1-00019 [F] 962 WSJ910827-0095 Technology & Medicine: Nissan Says Its Electric Car Reduces Battery-Recharge Time to 15 Minutes Nissan Motor Co. said it has built an electric car that can be fully recharge..... [F] 960 FT944-18127 FT 05 OCT 94 / Business and the Environment: Power to the petrol — John Griffiths finds battery cars cannot keep up In 2015, most cars will still be powered by conventional</p>
Search Database	Exit

Figure 2: Interactive interface: main search screen



Figure 3: Interactive interface: full record display

Xerox TREC-5 Site Report: Routing, Filtering, NLP, and Spanish Tracks

David A. Hull* Gregory Grefenstette* B. Maximilian Schulze*
Eric Gaussier* Hinrich Schütze† Jan O. Pedersen†

*Rank Xerox Research Center
{hull,grefen,schulze,gaussier}@grenoble.rxrc.xerox.com

†Xerox PARC
schuetze@parc.xerox.com

†Verity Corporation
jpederse@verity.com

1 Introduction

Xerox participated in TREC 5 through experiments carried out separately and conjointly at the Rank Xerox Research Centre (RXRC) in Grenoble and the Xerox Palo Alto Research Center¹. The remainder of this report is divided into three sections. The first section describes our work on routing and filtering (Hull, Schütze, and Pedersen), the second section covers the NLP track (Grefenstette, Schulze, and Gaussier), and the final section addresses the Spanish track (Grefenstette, Schulze, and Hull).

2 Routing and Filtering

For the routing and filtering tasks, Xerox continues its work on using statistical learning algorithms to develop optimal query profiles. The goal is to take advantage of the large number of evaluated documents available in the TREC context to build a more accurate model of each topic. There are a number of challenges in applying traditional learning algorithms to the routing/filtering problem, including a large and poorly defined feature set and a very low density of positive training examples. In this section, we describe the Xerox approach to document routing and filtering, concentrating on the differences from our experiments last year. Readers are encouraged to refer to the Xerox TREC-4 site report for more details on our work last year [11].

For the moment, Xerox does not have a specially designed text routing/filtering system. Instead, we rely on a traditional information retrieval system to simulate the text filtering problem. While this approach does not allow us to measure the time/space efficiency of our algorithms in a real-time system, it does provide accurate estimates of system performance according to precision-recall and utility measures. Xerox uses the Text Database System (TDB) developed at Xerox PARC [4] for parsing, tokenizing, stemming, and stop word removal. Indexed *terms* include single words as well as adjacent word pairs (not including stop words). Unlike last year, no document segmentation was performed for TREC-5.

¹Jan Pedersen was formerly at the Xerox Palo Alto Research Center.

2.1 Preliminary Data Reduction

Xerox uses a three step filtering process. First, an expanded query is created by taking the vector sum of the original query and all relevant documents in the training set. Non-relevant documents are ignored. Next, all available documents are ranked according to their similarity to the expanded query and the top 2000 documents are selected to define the *local region* for that query. All unevaluated documents in the local region are assumed to be not relevant. This preliminary filtering step serves several functions; the local region has a higher density of relevant documents and the non-relevant documents selected for training are those which are most difficult to distinguish from relevant documents, which should increase the discriminatory power of the system. Note that while we use the full document collection in these experiments, there is no reason why the local region could not be defined only in terms of the evaluated documents for a particular query. The only advantage of our approach is that it makes it easier to determine a threshold which defines when test documents should belong to the local region.

In the second step of the filtering process, an optimal feature set is determined for each query by selecting among the large number of potential terms available in the local region. We begin by ordering all terms according to their score based on the binomial likelihood ratio test statistic [5], with relevant and non-relevant documents representing the two classes. This statistic assigns a high score to documents whose probability of occurrence differs significantly among the two classes. We then filter out all terms with scores less than 5.0 (which corresponds to a p-value of 0.025 according to the chi-square test statistic), those which occur in fewer than five relevant documents, and those which assign a higher probability of occurrence to the non-relevant class (negative predictors). In preliminary experiments, we found no advantage to using negative predictors in our classifiers. This may be because the domain of non-relevant documents is so broad that it is dangerous to choose predictors based only on the local region. Of the remaining terms, the top 200 are selected to serve as the feature set for the statistical classifiers, and a reduced profile is created for each document in the local region. The binomial likelihood ratio test (LRT) is different from the chi-squared test that we used at TREC-4, even though it has the same reference distribution (chi-square). It seems that the LRT makes more realistic estimates of the importance of low frequency terms.

2.2 Statistical Learning of the Logistic Model

In the third and final step, the reduced profiles are sent to a statistical learning algorithm. In previous experiments, we have examined a number of different classification techniques for the routing problem, including logistic regression, nearest neighbors, linear discriminant analysis, and neural networks [14, 11]. At TREC-4, we found that a combination of techniques achieved the highest level of performance, while a single-layer neural network based on the logistic model was the best individual classifier. For TREC-5, we decided to concentrate on the logistic model and compare several different strategies for reducing overfitting,² which is probably the biggest barrier to good performance using this model. The logistic model produces explicit estimates of probability of relevance by applying the logistic transformation $g(\pi) = \log(\pi/(1 - \pi))$ to the output of a linear classifier. More details on logistic regression and the logistic model can be found in [8, 3].

While the two preliminary steps have substantially reduced the document and term set, previous experiments have shown that logistic regression still leads to overfitting [14]. One possible approach is to use an early stopping rule. Fitting the logistic model is an iterative process where parameter estimates gradually converge to the optimal solution. Unfortunately, the optimal solution for the training set is generally not an optimal model of the underlying population. We can come up with a better solution by estimating performance during convergence using cross-validation and stopping

²Overfitting is defined as improving performance on the training data while reducing performance on the test data. It is the result of techniques with high descriptive power which fail to generalize sufficiently from training data that is only an approximate model of the true underlying distribution.

when performance on the validation set begins to decrease. We have had good success in the past using an early stopping rule in conjunction with a neural network to attack the overfitting problem, but there are other possible solutions. For TREC-5, we examine three other techniques to reduce overfitting: regularization, variable selection, and non-negative logistic regression.

We can attack one obvious source of overfitting by taking advantage of how the feature set for the training data was selected. Since all terms in the reduced set are chosen to have a large positive association with relevance, it is fair to assume that any negative parameter estimates for these terms are not accurate. In an ideal world of term independence, this would never happen, but in practice, strong term correlation patterns may lead to negative estimates for certain parameters. Therefore, simply forcing all parameter estimates to be non-negative should lead to a more accurate model. We define this technique as non-negative logistic regression, and implement it in a simple fashion. After every five iterations of logistic regression, all terms with negative parameter estimates are removed from the model (which is equivalent to setting the estimate to zero). This process is continued until there are no negative parameter estimates. There are more principled ways to solve this problem based on the use of constrained optimization, but our heuristic approach seems to work well enough in practice. This simple step alone significantly improves the performance of the logistic model.

Regularization is a well-known statistical technique for improving the estimation of an overspecified model by biasing the estimated parameters towards a fixed value. In our case, we choose to use a simple probabilistic independence model as the fixed model. This approach can also be viewed as method combination, for it essentially boils down to taking a weighted combination of the fixed and fitted models. At TREC-5, we used this technique in conjunction with non-negative logistic regression.

We already applied variable selection in stage two of our algorithm to reduce the full feature set to roughly 200 selected terms that have a strong positive association with relevance. However, the number of selected terms is still large compared to the number of positive training examples, so further selection could be valuable to produce a more general model. One possible approach would be to use the same technique again but select an even smaller number of terms. However, the original algorithm has the key flaw that terms are selected independently of one another. A better strategy would select new terms that are the most helpful given the terms that are already in the model. This can be accomplished by using a backward stepwise variable selection technique in conjunction with logistic regression. Backward means that the model starts with all terms and terms are gradually removed in inverse order of their contribution to the model. Stepwise means the variables are removed one (or a few) at a time and the value of each remaining term is reassessed at each step in the process. We use a standard approach to model selection which evaluates each variable subset according to the AIC (Akaike Information Criterion) to decide which variables to remove and when to terminate the process. This approach bears strong similarity to the one adopted by City University at TREC-3 and TREC-4 [6], except that they use a forward selection process and optimize performance with respect to average precision directly.

Unfortunately, we could not fully test all the presented techniques within the context of TREC-5. Our three submitted runs all use the same preliminary data reduction steps described in the previous section, differing only in the learning algorithm. The first run applies a simple probabilistic independence model to the selected terms and is designed to serve as a baseline. No learning, other than the probabilistic term weights, is incorporated into this run. This model is not optimal because the terms are far from independent. Our second run is a linear combination of 0.7 times the probabilistic run plus 0.3 times the estimate from non-negative logistic regression. The parameter 0.7 was selected because it optimized performance over a previously analyzed query set and was left constant for all queries. The final run used the variable selection technique described in the previous paragraph.

2.3 Summary of the Xerox Algorithm

We provide a simple summary below of three stage training and testing process for our routing/filtering algorithm. The training process:

- Determine 2000 nearest documents to expanded query (local region).
- Select up to 200 important terms based on LRT statistic.
- Apply variants of logistic regression to determine linear classification rule for the local region.

The testing process:

- Select documents that exceed threshold for local region.
- Build reduced profile based on important terms.
- Score document according to classification rule.
- Routing: add document to priority queue of test documents.
- Filtering: accept or reject document based on filtering threshold.

The reader will note that we have made no distinction up to this point between the routing and filtering task. This is because our algorithms work identically in both cases, since logistic regression allows us to estimate the probability of relevance for each incoming document, and a simple transformation of the utility function allows us to describe the filtering threshold as a probability. The only difference comes after the document is scored. In routing, the document is added to a priority queue which ranks it relative to the other test documents, while in filtering, it is accepted or rejected by comparing its score to the probability threshold(s).

2.4 TREC-5 results

Measure	Prob	RNN-Logit	BStep-Logit
avg.prc	0.171	0.177	0.175

Table 1: Average uninterpolated precision for the 3 Xerox routing runs: probabilistic term weighting (Prob), regularized non-negative logistic regression (RNN-Logit), and backward stepwise logistic regression (BStep-Logit).

The Xerox TREC-5 routing results are presented in Table 1. Obviously, these results do not represent a triumph for our system, given that they are less than half the scores we received with similar runs last year. After analysis of the results, we found that the preliminary filtering step (selecting the local region via query expansion) failed in a dramatic fashion. To give you some idea of the magnitude of the problem, consider that only 49.6% of the relevant documents appeared in the top 2000 test documents (compared to 88.8% for TREC-2 and 94.2% for TREC-3) when computing similarity with respect to the expanded query.

Our query expansion technique is very simple, merely take the vector mean of the query and all relevant documents, so it is surprising to see such a difference from previous years. Further analysis reveals that the stop list (!) is responsible for the problem. We reduced the stop list from hundreds down to only a few entries in order to catch some important pairs that we had been losing in previous years. This added a large number of spurious terms and pairs which turned out to have a dramatic impact on expansion performance. Conventional wisdom in IR is that the selection of terms for removal has only a small impact on system performance, and this has always been the case in our previous experiments, so we thought we could manipulate the stop list with impunity. Obviously,

when indexing term pairs in the context of the routing problem, one has to be a lot more careful. After running the query expansion with a more substantial stop list, we found the coverage on the test set jumped to about 78% (and could move higher with more careful selection). We were not able to run the rest of the routing experiments in time for this paper.

Given the low coverage of relevant documents in the test set, we are reluctant to make any conclusions about the relative performance of the learning algorithms from the scores in Table 1. While average uninterpolated precision depends heavily on good recall, the same is not true for the utility measures used in filtering this year. Therefore, our filtering performance is much more competitive with other TREC-5 groups, although the system still probably suffers to some extent from the low recall problem.

3 Natural Language Processing and IR

For the Natural Language Processing (NLP) task, Xerox attempted to test whether linguistically motivated light parsing would improve retrieval results over the classic IR approximation to noun phrase recognition. This can be seen as a large-scale reproduction of Joel Fagan's experiments with CACM and CISI [7], in which he compared using positionally derived phrases to syntactically derived noun phrases as additional indexing terms.

3.1 Approximations to NLP in IR

In order to capture linguistically similar documents, Information Retrieval has always used approximations to some levels of natural language processing: tokenisation, for example, involves general knowledge of how words are formed; stopwords involves syntactic knowledge of what words act as function words; and stemming involves morphological knowledge about word formation. Recognition of one of the most common syntactic structures in English, the compound noun phrase, has long been approximated by considering all pairs of contiguous words between stopwords as additional indexing terms[12].

These approximations have the advantage of being relatively simple to implement, requiring little data, and resulting in fast treatment. Though tokenisation involves keeping state information while reading non-space characters, stopwords and stemming is essentially table lookup, and binary phrases involve only contiguous word pairs.

Natural Language Processing, on the other hand, involves larger structures than those considered in IR, and additional time is required to access those structure during text processing. In addition to more simple word tokenisation, the input text must also be tokenized into sentences for most steps in NLP. The context of the entire sentence is used in part-of-speech disambiguation, a necessary step for proper lemmatisation³ and for subsequent parsing. Within each sentence, more complicated structures such as complex noun phrases and verb phrases are recognized by NLP and relations within and between these phrases are extracted as additional index terms for information retrieval. This structuring of the sentence involves more work than in the classic Information Retrieval approach, but it allows the recognition of syntactic relations between non-contiguous words, such as between verbs and nouns separated by intervening words. We wanted to test whether proper recognition of syntactic word dependence would eliminate noise caused by chance co-occurrence of words and reduce silence by recognising non-contiguous syntactically related word pairs.

3.2 Non-NLP runs

In order to see if NLP provides any substantial gain over non-NLP methods, Xerox created classic Smart runs with and without phrases as baselines. These are the non-NLP runs described below.

³Lemmatization is the reduction of inflected words to their root forms, like *thought* as a verb to *think*. Though rarely the case in English, in most other languages, verbal forms and nominal forms lemmatize differently.

Original Text (non-stopwords in boldface):

Where and for what purpose is **scuba diving** done professionally?

Stemmed indexed terms:

purpos **scub** **dive** **profess**

Additional phrasal indexing terms (found between stopwords):

dive_scub

Figure 1: Classic IR approximation to phrasal indexing. The new term is composed of the stemmed versions of the contiguous now-stop words. In addition, the order of the elements is normalized into alphabetical order [12].

The NLP runs use a finite-state light parser[10] to extract syntactic word dependencies. The NLP runs using these extracted pairs as additional index terms are compared against the same baseline.

3.2.1 Phrases in Classic Information Retrieval

The classic phrase⁴ constructing method used in information retrieval systems like SMART is to create a new indexing term from contiguous pairs of non-stopwords[1]. See Figure 1.

SMART allows these new terms to be added into a distinct indexed field, the weighting of whose elements depends on the original document length. In this way, adding in extra terms does not make the document seem longer to the weighting procedure⁵, and thus the original uniterms (like **purpos**, **scub**, **dive** and **profess** in Figure 1) have the same weights as the word without the additional phrase.

The addition of these simply-derived phrases has consistently shown to improve retrieval[2]. This improvement has been reproduced in our implementation of this classical strategy.

As an efficiency issue with regards to space, the inclusion of phrases in the indexes greatly increases the number of index terms. Though much rarer, there are many more phrases than individual stems. Cornell [2] reports that they use as index items only those phrases that appeared at least 25 times in the initial TREC1 document set. Here for the Xerox nonNLP and NLP experiments, we retained any phrase appearing in the query as an index term. Since the weights of the phrases do not depend on the existence of any other phrases⁶, this allows us to limit the number of terms indexed without altering the theoretical results.

We will now describe the runs that we have performed for the NLP track of TREC 5. A recapitulative table is given in Figure 2. The table indicates results for TREC topics 251–300, run over Disk 2 data, 250 MB of Wall Street Journal text. Seventeen topics had 4 or fewer relevant documents in this set. A change in the relevance position of one or two documents in such queries can significantly alter the average precision of the entire run. For example, if a query has only one relevant document, if that document moves from second place in the ranked list of returned documents to the first place, then there is a 50% gain in precision, which even when averaged over 50 queries can still modify the total precision of a run by 1%. In order to eliminate the effect of these queries, Figure 2 also indicates results excluding these topics under the heading “33 topics.”

⁴Here *phrase* corresponds to a an index term derived from more than one word.

⁵In the normal case, the weight of each term in a document decreases as the number of index terms increases.

⁶Their weights only depend on their frequency throughout the database, and the length in words of the original documents.

<i>RUNS</i>		All 45 Topics				33 Topics (> 4 rel docs)			
		Avg		R		Avg		R	
<i>NonNLP</i>		Prec		Prec		Prec		Prec	
stemming	xerox_nlp1	0.200	—	0.206	—	0.189	—	0.228	—
+contig pair	xerox_nlp2	0.215	7%	0.217	5%	0.196	4%	0.228	0%
+wind pair	window	0.187	-6%	0.219	6%	0.158	-16%	0.216	-5%
<i>NLP</i>									
nlp	mwe	0.231	15%	0.248	20%	0.202	7%	0.240	5%
contig&nlp	xerox_nlp4	0.228	14%	0.246	19%	0.198	5%	0.237	4%
<i>Manual Correction</i>									
nlp	xerox_nlp5	0.232	16%	0.249	21%	0.204	8%	0.241	6%
contig&nlp	xerox_nlp6	0.230	15%	0.247	20%	0.200	6%	0.239	5%
<i>Short Queries</i>									
contig&nlp	xerox_nlp3	0.140	-30%	0.154	-25%	0.147	-22%	0.172	-25%

Figure 2: Comparative table of results Xerox's TREC 5 NLP track runs. **Stemming** is the baseline, a classic SMART run. The phrasal baseline is given by **+contig pairs** which, in addition to stems, includes additional index terms derived from sorted stemmed pairs of contiguous non stopwords. **+wind pair** includes pairs derived from a window of 10 non stopwords around each stem; **nlp** includes stems and sorted stemmed pairs of syntactically related words found by a light parser; **contig&nlp** includes the union of pairs derived from the light parser and from SMART contiguous pairs. The two runs under the heading *Manual Correction* underwent manual elimination of unwanted pairs from the queries. *Short Queries* is an automatic run using pairs derived from the parser and the SMART contiguous non stopword method, but only using the description field of the query. All other runs also use the "Narrative" field. The results given under the column labeled **33 Topics** give a truer picture, since topics with less than 5 relevant documents, whose results skew performance, are not considered.

3.2.2 Baselines: `xerox_nlp1` and `xerox_nlp2`

Our first non-NLP baseline is a classic SMART run, with stemming turned on. Documents were indexed using the “lnc” and queries using “lrc” weighting schemes⁷.

The run `xerox_nlp2` is an implementation of SMART phrases.

Since adding in new phrases means that words in the phrases are accounted for twice, once in the term and once as individual words, a common practice is to downweight the extra phrases. After a period of testing on former TREC queries, we found that multiplying the additional terms by 0.5 gave the best results.

We were not interested in extensively modifying SMART code, so we decided to build additional fields for each query and document that would contain phrasal indexes derived from other indexed fields. SMART provides an indexing method in which terms from such a field do not lengthen the document for weighting purposes, so the number of extra unused terms added does not affect retrieval performance of the individual words. Still, adding in new phrases means that words in the phrases are accounted for twice, once in the term and once as individual words. The common practice is to downweight the extra phrases. After a period of testing on former TREC queries, we found that multiplying the additional terms by 0.5 gave the best results. To reduce the size of these fields, we only retained, as document phrases, phrases that were also derivable from some query among 251 to 300. Although, this is not the approach that would be taken in a commercial system, it provides the same theoretical results as if all possible contiguous word pairs were indexed for each document.

3.2.3 Non-contiguous non-NLP pairs: window

Another run (labeled “+wind pair” in Figure 2, but not submitted to TREC evaluators) extends the idea of creating additional indexing terms from pairs of non stopwords. Instead of just indexing contiguous pairs, any pair within a window of 10 non-stop words was created as an indexing term. This run was performed in order to test the idea that any gain from syntactic recognition of non-contiguous pairs could be approximated by simply widening the window from which pairs were drawn. In order to be complete, we should have run this test with window lengths of 2 to ten.

3.2.4 Automatic NLP runs: `mwe` and `xerox_nlp4`

The “mwe” run⁸ includes index terms made from pairs of words sharing some syntactic dependency in indexed text. Using a light parser[9, 10], a shallow syntactic analysis was made of the queries and documents, and any pair of words found to be in one of the following relations were extracted: subject-verb, verb-direct object, verb-adjunct, noun modifying noun, adjective modifying noun, adverb modifying verb. Pairs involving a SMART stopword were eliminated. The words in remaining pairs were stemmed using SMART’s stemming algorithm, and alphabetically sorted pairs were constructed, as shown in Figure 3.

The “mwe” preprocessing⁹ that produces an extra field in each document before classical SMART indexing consists of the following steps: (i) mapping SGML-coded characters to ISO, (ii) tokenisation (iii) morphological analysis (iv) part-of-speech disambiguation (v) verb group recognition (vi) noun group recognition (vii) marking verb group aspect and head (viii) marking noun group heads

⁷Actually, instead of using log of document frequency, we prefer using square root of document frequency. This requires the modification of one line of SMART code.

⁸This run was not submitted for evaluation, since the strategy used in `xerox_nlp4` had performed better on our test corpus.

⁹This entire process (a sequence of binaries, shells, and *gawk* programs) performs at about 360 words/second or 2.4 Kbytes/sec on a SPARC Ultra 1 with 64 Mbytes of memory. Preparing the whole NLP track (250MB), before SMART indexing, takes about 36 hours. 2 additional hours were needed by another *perl* program to restem the extracted pairs according the SMART stemmer, and to eliminate pairs not appearing in the queries. Indexing the preprocessed data on a SPARC Ultra 2 with 512 Mbytes of memory takes about 15 minutes.

Original Text:

Where and for what purpose is scuba diving done professionally?

Stemmed indexed terms:

purpos scub dive profess

Parser-derived relations:

do professionally

dive professionally

scuba do

scuba dive

Additional phrasal indexing terms (after stemming and stopword removal):

dive_scub dive_profess

Figure 3: Light parsing version of indexing pairs. In addition to stemmed indexed single-word terms, the same as SMART classically produces, additional index terms are derived from the parser output. The parser produces binary relations between lemmas. If the pair contains a stop word, it is eliminated. Otherwise, the lemmas are stemmed according to SMART's stemmer, sort in alphabetic order and joined by an underscore to produce the new index term.

(ix) extraction of syntactic dependency relations (x) elimination of syntactic dependency relations involving the word 'document' (xi) reformatting of indexing pairs (xii) stopword elimination.

The "xerox_nlp4" run combined the pairs extracted by the NLP method outlined in the last paragraph with those extracted by the classical SMART method. For example, from the phrase "to combat alien smuggling", the SMART phrase builder will generate "alien_combat" and "alien_smuggl" while the light parser will only return "combat_smuggl" and "alien_smuggl". The union provided by both methods produces all three index-terms. This run was also performed to see if errors in the parses, notably silence, could be stopgapped by including the simple contiguous pairs of classical SMART phrases¹⁰

3.2.5 Manual NLP runs: xerox_nlp5 and xerox_nlp6

The additional indexing phrases produced for the topics from runs "mwe" and "xerox_nlp4" were manually edited to remove pairs derived from incorrect syntactic analyses and containing query metalanguage to produce the topics used in runs "xerox_nlp5" and "xerox_nlp6." Only removal was performed, there were no additions or corrections.

For example, in topic 251, the description reads: "Documents will report the exportation of some part of U.S. Industry to another country." The light parser extracted "countr_industr" as a pair. The human filterer judged this to be an incorrect parse and eliminated the phrase.

3.2.6 Discussion of Results

The results given in Figure 2 show an improvement in Average Precision and in R-Precision using all the multiword generating techniques, except the "window" which includes all possible non stopword combination generatable over 10 word windows.

The automatic nlp run "mwe" performs twice as well as the classic SMART contiguous pair technique¹¹. One might argue that the improvement comes from the fact that the parser, by creating

¹⁰For example, the query title "Combating Alien Smuggling" yielded no dependencies from the version of the parser used in TREC5. "Alien_smuggl" was returned from the Description section of the topic which contained "to stop the smuggling of aliens" but the index term "combat_smuggl" was not found for the run "mwe" for query 252.

¹¹Albeit at a cost of 36 hours processing time.

new terms from syntactically dependent words, creates additional local non-contiguous terms. The “window” run, while not invalidating this hypothesis, does weaken it somewhat since the window includes all the pairs found by the contiguous pairs and the parsing technique, and then some, yet it performs worse than simple stems. This shows that adding in all possible local pairs is not good, and suggests that light parser making some of the right choices about what should be combined.

The Figure 2 presents results averaged over all 45 topics which had 1 or more relevant documents, showing a significant improvement, and then over the 33 topics for which at least 5 relevant documents exist in the collection. The improvements found throughout the runs in this latter set are more modest. We decided to present these results after examining the result of Topic 253 “Cryonic suspension services.” According to assessors, this document only has one relevant document in the WSJ set used for the NLP track. This relevant document is ranked in second position by the runs using classic SMART and classic SMART with phrases, but becomes the first document in the NLP runs. This gives a 100% improvement for this query, which, averaged over the 45 topics, adds a 0.011 absolute improvement to the Average Precision, which would account in itself for 5% improvement in the whole run (given a 0.200 baseline as in Figure 2). Since the NLP runs show a 15% improvement in Average Precision for all 50 topics, this means that 33% of the improvement comes from the simple permutation of one document in one query. This is clearly unreasonable to have such a small change play such a large part in evaluation. To negate such wild effects, then, we removed the 17 topics that had 4 or fewer relevant documents from consideration, as shown in the column labeled “33 Topics (>4 relevant)” in Figure 2.

Analysing the reasons for this improvement means going into the documents and extracting what matched between query terms and competing runs. We developed an analysis tool that allows us to compare the index matches between a given query and a given document over a number of runs. For topic number 253 and the relevant document numbered 19058 by SMART, comparing classic SMART phrase indexing (run “xerox_nlp2” in Figure 2) and light parsing (run “mwe”) gives us the following chart, in which the runs are labeled 0 and 1, and in which the column “word” gives the index terms that matched in query and document with their weights (a dash in the weights means that the term was not present in that run):

Term Table for Query 253 and Document 19058
 0 ... xerox_nlp2
 1 ... mwe

word	0	1
body	0.00717	0.00717
body_preserv	-	0.01992
cure	0.01004	0.01004
freez	0.04717	0.04717
industr	0.00361	0.00361
long	0.00413	0.00413
nitrogen	0.01867	0.01867
preserv	0.01168	0.01168
report	0.00262	0.00262
servic	0.00295	0.00295
stor	0.00503	0.00503

From this table we see that the improvement in the runs come from the addition of the index term “body_preserv” found by the light parser in both topic 253 and in document 19058 (WSJ900917-0124). In that document, which is really about freezing CD’s, we find the sentence:

Cryogenics is best known as the method used to preserve the bodies of dead people until cures for their illnesses are discovered.

The light parser correctly extracts the syntactic relation between “preserve” and “bodies” which stems to “body_preserv.” But the topic 253 contains only the line:

Cryonics suspension is the practice of quick-freezing a human body immediately upon death and its preservation in a nitrogen environment for possible resuscitation in the event that a cure is found for the cause of death.

The parser here has extracted “body_preserv” because it has incorrectly recognized the scope of the preposition “upon” and had analyzed the sentence as if it read “a human body ... upon preservation” which although semantically correct here is only recognized by chance since the parser does no analysis of pronouns and can not reattach “its” to “body.”

Topic 253 is not included in the “33 Topic” column of Figure 2 since it only has one relevant document. A better explanation of the NLP runs modest improvements in this column comes from Topic 283 “China trade policy.” For this query, document 36340(WSJ910617-0166) jumps from 21st position to 11th because it contains two terms that the simple contiguous word phrase-finding method finds neither “chin_polic” nor “chin_trad”:

Term Table for Query 33 and Document 36340

0 ... xerox_nlp2

1 ... mve

word	0	1
chin	0.07687	0.07687
chin_polic	-	0.01337
chin_trad	-	0.02414
comp_us	0.01112	0.00988
compan	0.00230	0.00230
consum	0.01175	0.01175
cost	0.00285	0.00285
effect	0.00502	0.00502
foreign	0.01202	0.01202
polic	0.00848	0.00848
produc	0.00788	0.00788
trad	0.00991	0.00991
u.s	0.01893	0.01893
union	0.00430	0.00430
work	0.00445	0.00445

The light parser can extract “china_trade” from the document’s “...China’s main trade” while the contiguous pairs method will not.

3.3 Conclusion of NLP track

The principal approximation to linguistic knowledge used by classic English information retrieval is stemming. The stemming algorithms developed for English over the past thirty years have become very close to capturing much of inflectional and derivational morphology of English. Using stemmers provides a significant increase in information retrieval performance. A minor improvement is added by the approximation of phrases, by joining together contiguous non-stop words. From our results, it would seem that light parsing can slightly improve this phrasal indexing, but at a cost, of course, of a much heavier linguistic apparatus, and lengthened time of preprocessing. Nonetheless, we are optimistic that this approach will prove useful in the long run for a number of reasons: (1) For non-English languages more work is being done on linguistic analysis than on information retrieval. This implies that morphological analysers for these languages may largely outperform simple stemming routines that have undergone the same maturation time as English stemming routines. (2) As machine become more powerful, the preprocessing times will continue to fall, making more complicated text analysis more economically feasible. (3) Robust parsing is progressing in the variety and accuracy of structures recognized. If contiguous structures improve retrieval, the recurring but separated structures should also, as Figure 2 supports.

4 Spanish Track

For the Spanish track, Xerox continues to test its linguistic analysis tools in the context of information retrieval. At TREC-4, we examined the performance of a linguistically motivated lemmatiser for Spanish stemming and found that it consistently improved performance (and is slightly more effective than the English version). For TREC-5, we use a Spanish part-of-speech tagger and pattern matcher to identify and index noun pairs as a linguistic alternative to simple adjacent pairs that are often used in information retrieval. We also conduct our first experiments on English/Spanish cross-language retrieval using English versions of the query and an English-Spanish bilingual dictionary.

4.1 Xerox Spanish runs

Our baseline run (xerox-spS) uses Xerox Spanish morphological tools as a linguistic alternative to classical stemming algorithms. In this process, the text is tokenized, tagged, and then the tagged version is passed to the lemmatizer [13], which means that terms are unambiguously reduced to their root forms. In general, this approach is robust and accurate, with a few minor exceptions: tagger errors and terms which are missing from the lexicon. The former may result in terms being reduced to the wrong root form while the latter means that some terms remain unreduced. Accents are removed to help recognize and conflate accented and unaccented version of the same terms. There are very few words in Spanish that have different meanings depending on their accentuation, so this step does not introduce much ambiguity. However, it is not a general solution for all languages. Since these experiments were conducted, we have developed versions of our morphological tools which will conflate accented and unaccented versions of a term if there is no ambiguity. The resulting text is indexed and documents are retrieved in the traditional fashion (using a classic SMART system).

The two noun-phrase runs (xerox-spP and xerox-spD) use indexed phrases which are built by combining nouns with adjacent non-stop words, ignoring the prepositions: ‘de’, ‘para’, and ‘a,’ and their composed forms. Xerox-spP is run on the long version of the query while xerox-spD uses the description field only. For example the sentence: “Senalo que en ese estudio tecnico tambien debera establecerse la factibilidad de realizar una nueva convocatoria para otorgar mas concesiones para ese servicio.” would generate “estudio.tecnico factibilidad_realizar convocatoria.nuevo convocatoria.otorgar” in addition to the lemmatised word forms. As with the English phrases described in the section on the NLP track, phrases are ignored for document length determination, and their term weight is halved to reflect the fact that phrases also match on the component terms.

For our cross-language experiment (xerox-spT), we use the English versions of the full Spanish topics that were supplied by NIST this year. Our system applies tagging and lemmatization [13] to reduce all English terms to their root forms. The terms are then automatically looked up in a general-language bilingual dictionary and a translated query is generated by replacing each English term with all of its Spanish translations. The newly constructed Spanish query consists of the union of all of these translation equivalents and is evaluated in the traditional fashion. We have been recently experimenting with a weighted boolean model for cross-language information retrieval that tends to work much better than the vector space model, but this research was not completed in time for our submission. We hope to explore this model further at TREC-6.

4.2 TREC-5 Spanish results

Unfortunately, our FTP session was interrupted while we were in the middle of downloading the AFP data for testing, and we were unaware that this had happened. As a result, we ended up running our Spanish experiments on only three-quarters of the data, so the runs submitted to TREC-5 are not accurate indicators of the true performance of our system. The biggest impact of this problem is to reduce our recall, but precision may also have been affected. The results of running over the full data is given in Figure 6.

Original Spanish Query SP75:

<title> Heroin in Latin America
<desc> Description: Hay un mercado para la heroína producida en América Latina
<narr> Narrative: Con las grandes incantaciones de la coca enviada a los EEUU, ciertos de los productores latinos han vuelto a la producción de heroína.

TREC-supplied English version of SP75:

<title> Heroin in Latin America
<E-desc> Description: Is there a market for the heroin produced in Latin America?
<E-narr> Narrative: With the massive seizures of cocaine shipped to the U.S., certain Latin American producers have turned to the heroin trade.

Re-translated Spanish version of SP75:

<desc> valor en el mercado, precio de mercado, plaza del mercado, mercado, los puestos del mercado, lonja, informe de mercado, fuerzas, economía de mercado, cuota de mercado, creador de mercado, comercializar, bolsa de valores, analista de mercado heroína, heroinomano tener, surtir, sacar, realizar, prolongar, productos alimenticios, producir, presentar, poner en escena, montar, fabricar, encargarse de la puesta en escena, encargarse de la producción, encargarse de la dirección, dirigir, devengar, dar, causar, ... (126 other words)
agriar, adelantar, acostarse, acortar, abrir, (hacer) girar heroína, heroinomano traspaso, traspasar, ruta comercial, publicar las estadísticas de la balanza comercial, publicación especializada, oficio, jugador traspasado, intercambiar, industria, hacer un cambio, guía de fabricantes y comerciantes, feria comercial, explotar, entregar como parte del pago, el desequilibrio de la balanza comercial, déficit en la balanza comercial, descripción comercial, comercio, comerciar, capitalizar, barrera arancelaria, acuerdo comercial
</desc>

Figure 4: Cross-Linguistic Baseline: Each English word is replaced by all translations found in general language bilingual dictionary.

<i>Spanish Runs</i>		Avg Prec	R-Precision
inflectional stemming	xerox-spS	0.295	0.338
noun phrase pairs	xerox-spP	0.301	0.350
" (short Queries)	xerox-spD	0.275	0.326
translated Eng-Span	xerox-spT	0.116	0.155

Figure 5: Spanish results over three-quarters of the data.

<i>Spanish Runs</i>		Avg Prec	R-Precision
inflectional stemming	xerox-spS	0.410	0.415
noun phrase pairs	xerox-spP	0.417	0.425
" (short Queries)	xerox-spD	0.361	0.384
translated Eng-Span	xerox-spT	0.160	0.188

Figure 6: Spanish results over full data.

References

- [1] Chris Buckley, Gerard Salton, James Allen, and Amit Singhal. Automatic query expansion using smart: TREC-3. In D.K. Harman, editor, *The Third Text REtrieval Conference (TREC-3)*. U.S. Department of Commerce, 1995.
- [2] Chris Buckley, Amit Singhal, and Mindhar Mitra. New retrieval approaches using smart: Trec4. In D.K. Harman, editor, *The Fourth Text REtrieval Conference (TREC-4)*. U.S. Department of Commerce, 1996. to appear.
- [3] Wm. S. Cooper, Aitao Chen, and Fredric C. Gey. Full text retrieval based on probabilistic equations with coefficients fitted by logistic regression. In D.K. Harman, editor, *The Second Text REtrieval Conference (TREC-2)*, pages 57–66. U.S. Department of Commerce, 1994.
- [4] Douglass R. Cutting, Jan O. Pedersen, and Per-Kristian Halvorsen. An object-oriented architecture for text retrieval. In *Conference Proceedings of RIAO'91, Intelligent Text and Image Handling, Barcelona, Spain*, pages 285–298, April 1991. Also available as Xerox PARC technical report SSL-90-83.
- [5] Ted Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74, 1993.
- [6] S.E. Robertson et al. Okapi at TREC-3. In D.K. Harman, editor, *The Third Text REtrieval Conference (TREC-3)*, pages 57–66. U.S. Department of Commerce, 1995.
- [7] Joel L. Fagan. *Experiments in Automatic Phrase Indexing for Document Retrieval: A Comparison of Syntactic and Non-Syntactic Methods*. PhD thesis, Cornell University, September 1987.
- [8] Norbert Fuhr and Ulrich Pfeifer. Probabilistic information retrieval as a combination of abstraction, inductive learning, and probabilistic assumptions. *ACM TOIS*, 12(1), Jan 1994.
- [9] G. Grefenstette. Use of syntactic context to produce term association lists for text retrieval. In *Proceedings of SIGIR'92*, Copenhagen, Denmark, June 21-24 1992. ACM.
- [10] G. Grefenstette. Light parsing as finite state filtering. In *Workshop on Extended finite state models of language*, Budapest, Hungary, Aug 11–12 1996. ECAI'96.
- [11] Marti Hearst, Jan Pedersen, Peter Pirolli, Hinrich Schütze, Gregory Grefenstette, and David Hull. Xerox TREC-4 site report. In D.K. Harman, editor, *The Fourth Text REtrieval Conference (TREC-4)*. U.S. Department of Commerce, 1996. to appear.
- [12] Gerard Salton, C. S. Yang, and C. T. Yu. A theory of term importance in automatic text analysis. *JASIS*, 26(1):33–44, 1975.
- [13] Anne Schiller. Multilingual part-of-speech tagging and noun phrase mark-up. In *15th European Conference on Grammar and Lexicon of Romance Languages*, University of Munich, Sept 19–21 1996.
- [14] Hinrich Schütze, David Hull, and Jan Pedersen. A comparison of document representations and classifiers for the routing problem. In *Proc. 18th Int'l Conference on R&D in IR (SIGIR)*, pages 229–237, 1995.

Term importance, Boolean conjunct training, negative terms, and foreign language retrieval: probabilistic algorithms at TREC-5

Fredric C. Gey, Aitao Chen, Jianzhang He, Liangjie Xu, and Jason Meggs

U.C. Data Archive & Technical Assistance (UC DATA)

University of California, Berkeley

e-mail: gey@ucdata.berkeley.edu

ABSTRACT

The Berkeley experiments for TREC-5 extend those of TREC-4 in numerous ways. For routing retrieval we experimented with the idea of term importance in three ways -- training on Boolean conjuncts of the most important terms, filtering with the most important terms, and, finally, logistic regression on presence or absence of those terms. For ad-hoc retrieval we retained the manual reformulations of the topics and experimented with negative query terms. The ad-hoc retrieval formula originally devised for TREC-2 has proven to be robust, and was used for the TREC-5 ad-hoc retrieval and for our Chinese and Spanish retrieval. Chinese retrieval was accomplished through development of a segmentation algorithm which was used to augment a Chinese dictionary. The manual query run BrklyCH2 achieved a spectacular 97.48 percent recall over the 19 queries evaluated before the conference.

1. Introduction

From the beginning of the TREC conference series, the UC Berkeley Text Retrieval Research Group has been developing probabilistic algorithms to retrieve full-text documents from collections as large as 1 million documents. The Berkeley approach has been 'bare bones', concentrating on fundamental algorithm features rather than a 'kitchen sink' approach of adding ad-hoc features (like passage retrieval or phrase discovery, etc.) merely because they add a modicum to performance. Our core approach has been to use the statistical technique of *logistic regression* to predict relevance as a function of statistical attributes of query terms common to both document and query. Logistic regression is comparable to other probabilistic approaches such as neural networks (Kwok 1996), inference networks (Turtle and Croft 1991) and 2-Poisson (Robertson and Walker 1994).

2. TREC-5 routing methodology

The Berkeley TREC-4 routing results were puzzlingly low, since our basic training seemed to show that we should have achieved better results than TREC-3. After the TREC-4 conference we reran our algorithms from TREC-3 on the TREC-4 data and found that they performed better than TREC-4 run Brkly12. The basic feature of the TREC-3 algorithm was to:

- Perform a χ^2 analysis to find terms which had the highest statistical dependence on relevance
- Choose all terms (for query expansion) where the χ^2 had probability value greater than 5 percent significance level. This produced between 300 and 4114 terms depending upon the query, with a mean query size of 1,357.

For TREC-4 we had believed that we could truncate the number of expansion terms to the 300 most important ones (in terms of the size of χ^2). This judgment turned out to be faulty -- a retrospective run using all expanded terms improved performance by 20% over the official Brkly12 run. It seems that an incremental evidence contribution from a large number of insignificant terms can be more important than the contribution from a few important terms.

Thus, for TREC-5 we returned to the concept of massive query expansion using the χ^2 discrimination measure to choose terms. This, using the TREC-3 formula for combination of evidence from terms, forms the method for our first routing run, Brkly13. The number of terms for TREC-5 routing varied from a minimum of 714 (topic 222) to a maximum of 3839 (query 82), with an average of 2032 terms over the 50 queries.

2.1. Boolean filtering for important terms

Despite the evidence that massive query expansion seems to yield our best performance, we have continued to be intrigued by the OKAPI results which depend upon choosing the 15 "best" terms for expansion and ignore the rest (Robertson, Walker, et.al. 1995). We wondered whether some use of these "best" terms might strip away noise documents retrieved by other terms. What we wish is to utilize those terms which have the most effective discriminating power on the basis of their relevance history.

For example, if we consider the TREC-5 topic 003 (which was also a TREC-4 topic):

Domain: International Economics

Topic: Joint Ventures

Description: Document will announce a new joint venture involving a Japanese company.

Narrative: A relevant document will announce a new joint venture and will identify the partners (one of which must be Japanese) by name, as well as the name and activity of the new company.

The following ranks the query terms according to their term absence logodds of relevance:

Important terms for TREC query 003		
Query	Term	Logodds
003	joint	-7.578775
003	ventur	-6.022744
003	japan	-5.706141
003	compan	-1.363744
003	produc	-0.790130
003	corp	-0.536089

From this table we conclude that the most important terms are *Japan*, *joint* and *venture*. The scale indicates that if the term *Japan* is absent from a document, the document has odds 8 times (3.73 times value of e) less likely to be relevant than if the terms *produc* or *corp* are absent from the document.

Retrieval performance by an official Berkeley TREC-4 Brkly12 entry for this query using probabilistic retrieval by a logistic regression equation (Gey et. al. 1995) produced average precision of 0.2852, below the median precision of 0.3765 for that query and less than half the best precision entry of 0.6781. However, if we restrict our ranking to only documents which contain the words *Japan*, *joint*, and *venture*, the performance increases to 0.6443, a substantial improvement and close to the best TREC-4 entry for this query. Further experiments, described in detail in (Gey, Chen 1996), showed that an intelligent choice of Boolean filters for all 50 queries of TREC-4, would have improved the overall precision of Brkly12 from 0.2163 to 0.3317, fifty-three percent higher than our official run.

2.2. Training on Boolean conjuncts

The use of the triples of Boolean conjuncts led us, in preparation for TREC-5, to the concept of training on Boolean "mintterms" for the top terms chosen for query expansion in routing. Mintterms are the elementary conjunct combinations of term presence or term absence when a Boolean query is transformed into its *disjunctive normal form*. The idea was to rank the mintterms on their historical relevance density, which would give their order as to how each subset of documents might produce the greatest number of relevant documents in rank order. As we ran retrospective runs on the TREC-

4 disks2 and disk3 collections we found that precision improved the more minterms we used, until, using the top 15 terms (32,767 minterms) we would achieve a staggering precision of 0.8048.

Of course this was too good to be true -- we had fallen into the trap of applying our training set to our training collection rather than a new collection. When we applied the minterm rankings and coefficients to another collection (disk1) , performance always decreased, no matter how many or how few minterms we used. It seems that while the relevance history of terms found in relevant documents has considerable predictive power, the Boolean combination of terms does not.

Boolean minterms experiments average precision		
Number of terms	retrospective	predictive
5	0.2985	0.3194
10	0.4935	0.2405
15	0.8048	0.1086

We are pursuing further failure analysis of these results.

2.3. Logistic regression in routing

After having spent several months modifying our software in pursuing the blind alley of Boolean minterms, and with three weeks left before the results were due, the Berkeley group needed to find another approach. We continued to believe that some account needs to be made which incorporates the contribution from the most important terms. After some thought we settled upon an algorithm which trains on the 15 most important terms (in order of historical logodds of relevance when the term is absent from the document) and the total evidence from all terms chosen for query expansion (i.e. the RSV of our baseline run Brkly13). Thus the formula for the Brkly14 run is as follows:

$$\log O(R | q_i, d_j) = c_0 + c_1 \log(tf_1 + 1) + \dots + c_{15} \log(tf_{15} + 1) + c_{16} weight + c_{17} \log DL$$

where tf_i means the term frequency of the i th term in the document, $weight$ is the logodds of relevance weights summed over all expanded query terms (the TREC-3 formula) found in the document, and finally DL_j is length of the document. This formula not only includes contributions from all terms, but adds an appropriate contribution for the 15 most important terms. Note that there are actually 50 regression equations, one for each query. This training-by-query was another improvement over TREC-4, where we produced one equation averaged over all 50 queries. Our training set was all relevant documents and a one-percent sample of non-relevant documents from the entire collection (not just the judgment set for that query). It is worth noting that the SPSS statistical package will solve all 50 equations in 15 minutes, as opposed to 15 minutes per equation using the S Plus statistical system.

2.4. TREC-5 routing performance

The Brkly14 run (average precision of 0.2601) performed 21 percent better than Brkly13 (average precision of 0.2156) in overall precision over the 45 queries, and 16 percent better over the 39 queries with more than two relevant documents. Brkly14 performed median or above in 35 of the 45 queries with 5 bests for their queries. If we compare TREC-5 to TREC-4 overall, it seems that the average median precision over all participating groups has dropped from .298 to .204, raising a question as to whether the target collection for TREC-5 was more difficult. This seems likely, since the distribution of relevant documents is highly skewed -- four queries (111, 142, 189, 202) account for 49.3 percent of all relevant documents for the 45 queries.

3. TREC-5 ad-hoc methodology

Berkeley did not spend much effort refining its approach to ad-hoc retrieval. The formula for ad-hoc has remained unchanged since TREC-2, when the concept of Optimized Relative Frequencies (ORFs) was introduced.

Berkeley's basic formula for prediction of logodds of relevance between a query Q and document D when there are M match terms in common between query and document, is

$$\log O(R | t_1, \dots, t_M) \approx -3.51 + \frac{1}{\sqrt{M} + 1} \Phi + 0.0929 M$$

where Φ is the expression

$$37.4 \sum_{j=1}^M \frac{Qtf_j}{QL+35} + 0.330 \sum_{j=1}^M \log \frac{Dtf_j}{DL+80} - 0.1937 \sum_{j=1}^M \log \frac{Ctf_j}{CL}$$

where

Qtf_j = number of times the j 'th term occurs in the query and QL = total number of all term occurrences in query (query length);

Dtf_j = number of times the j 'th term occurs in the document, and DL = total number of all term occurrences in document (document length);

Ctf_j = number of times the j 'th term occurs in the collection, divided by CL , the total number of all term occurrences in the collection (collection length);

M = number of distinct terms common to both query and document.

This formula was used, without modification, in Berkeley's Chinese retrieval experiments discussed below.

Since Berkeley's TREC-4 performance was only average on the extremely short queries introduced in TREC-4, we speculated that the contribution from query terms:

$$\frac{Qtf_j}{QL+K_q}$$

might be overwhelmed by the constant $K_q=35$. Indeed, Kwok has noted (Kwok, 1996) that the average query size for the 49 queries of TREC-4 was about 6 terms, while the average query size for the 50 queries of TREC-3 was 19 unique terms.

In order to test this, Berkeley ran a number of retrospective tests for both short and long queries to see if they could be improved upon. We trained on the short description queries from topics 151-200 on TREC DISK2 and tested on the short queries from topics 201-250. The results, summarized in this table:

TREC-5 Tests on query constant Average Precision for different values of K_q		
query constant	Average precision	R-Precision
10	0.1864	0.2442
15	0.1951	0.2507
20	0.1955	0.2506
25	0.1933	0.2494
30	0.1880	0.2498
35 (base)	0.1961	0.2511
40	0.1770	0.2384

show that results are not particularly dependent upon constant value unless it exceeds 40 or falls below 15. Since the best performance comes with $K_q = 35$ we retained this value for our TREC-5 experiments.

3.1. A cautionary tale of ad-hoc query expansion

In TREC-4 our official runs used a technique of query expansion whereby the top documents of a 'trial retrieval' are assumed to be relevant, and some of their terms are added to the query. Following TREC-4, Berkeley did a more systematic series of runs connected with query expansion. These runs, summarized in the table below, show that Berkeley would have been better served by making its official entry the run without expansion, which at 0.2945 was nearly 10 percent better than the 10 document/ 20-term expansion of the actual entry.

TREC-4 Brkly10 Ad-hoc Entry Average Precision for document/term expansions				
0.2945*	10 terms	20 terms	50 terms	100 terms
10 docs	0.2821	0.2660**	0.2335	0.1995
20 docs	0.2646	0.2533	0.2263	0.1857
30 docs	0.2589	0.2477	0.2138	0.1885

* - No expansion, ** - Brkly10 official entry

Moreover, the same is true of our TREC-4 automatic entry, Brkly9; the official entry using query expansion had a precision of 0.1388. Without query expansion, the average precision would have been 0.2001, more than 45 percent better than our official entry.

Berkeley abandoned this ad-hoc query expansion method for TREC-5, feeling that we have not yet mastered this art. Thus all TREC-5 ad-hoc runs use only the original (automatic or manually constructed) query terms, as do our Spanish and Chinese runs.

3.2. Manual query reformulation

Manual query reformulation has become the de-facto approach to enhancing retrieval for the ad-hoc problem. Berkeley has continued this tradition in TREC-5. Of course, the TREC committee decision to allow us to examine the top documents of a trial retrieval made this reformulation easier, although Berkeley used a combination of this and extra-curricular retrieval from a commercial news database in order to find additional terms.

3.3. Negative terms

During the course of TREC-4 Berkeley made a few cursory investigations into Boolean filtering with manually generated negative terms. A negative term, in a Boolean model, would be one which would exclude the document if the term appeared in it (i.e. the term would appear as AND NOT *term* in a Boolean query formulation). For TREC-4 we found negation to be entirely too rigid a condition.

Yet for TREC-5 there seemed to be queries for which some type of negation would be helpful. For example, retrieval using topic 292,

<num> Number: 292

<title> Topic: Worldwide Welfare

<desc> Description:

Identify social programs for poor people in countries other than the U.S.

<narr> Narrative:

To be relevant a document would identify a welfare program in a foreign country and explain how it works to aid citizens who have little or no income. It would include those who can't work because of a disability and people who have the extra burden of small children. The document should indicate how these people are supported or not supported. A relevant document should identify the source of the monies used to support such welfare programs.

retrieves many documents about the continuing controversy over welfare reform legislation in the United States.

Our speculation was that if "negative" terms such as 'Clinton', if they appeared in a document, the final weight would be reduced, the noise documents from the U.S. welfare system would be weeded out. Our final algorithm for this was to take the weight equations above and divide by the square root of the number of negative terms plus one. This seemed, prospectively, to work in that the first 10 documents of the Brkly17 run for this query were:

1. Welfare Overhaul Gets Final Touches In Senate
2. Senators Meet To Try To Find Welfare Plan Acceptable to Reagan
3. Surging Welfare Costs and the Struggle to Control Them
4. leading article: reforming usa welfare
5. Finance Committee Clears Bill To Overhaul Welfare System
6. An ecu for europe's poor: delors wants to put poverty higher on the agenda
7. Canada to unveil radical social security blueprint
8. Poor in the Country What The Presidential Candidates Propose
9. read clinton's lips: no more welfare: america
10. french senate approves state aid for companies cutting workers' hours

while the first 10 documents of the Brkly18 run were:

1. Canada to unveil radical social security blueprint
2. french senate approves state aid for companies clipping cutting
3. (no title: topic Reagan welfare reform)
4. benefits 'must target the poor'
5. New Study -- Most Homeless People Are Just Poor _ Not Mentally Ill
6. THE IMPORTANCE OF TWO-PARENT FAMILIES (House - April 28, 1994)
7. graduates 'happier on government unemployment benefits welfare dole than in stop-gap jobs'
8. one-day strike closes government unemployment benefits welfare offices
9. agency aids jobless graduates
10. Welfare Overhaul Gets Final Touches In Senate

Unfortunately, however, precision for Brkly18 was 0.0279, worse than Brkly17's precision of 0.0287 for topic 292. Overall precision over 47 queries was 0.2044 for Brkly18 and 0.2417 for Brkly17 runs without negative terms. This phenomenon held true for Spanish and Chinese as well. While we continue to believe in the viability of negative terms, how they should be combined remains elusive.

4. Berkeley ad-hoc results

In TREC-5 we had the luxury of submitting four runs, two automatic and two manual. In the following table we place the results of these four runs, together with the average of medians over all TREC-5 runs of that particular category;

TREC-5 Berkeley Ad-hoc Entries Average Precision for 47/50 queries				
Run number	47 queries		50 queries	
	Berkeley	All TREC-5	Berkeley	All TREC-5
Brkly15	0.1475	0.1529	0.1420	0.1437
Brkly16	0.2125	0.2026	0.2076	0.1905
Brkly17	0.2417	0.2314	0.2346	0.2174
Brkly18	0.2044	0.2314	0.1983	0.2174

Brkly15 - automatic run, short queries

Brkly16 - automatic run, long queries

Brkly17 - manual run, reformulated queries

Brkly18 - manual run, reformulated queries with negative terms

5. TREC-5 Chinese retrieval

Berkeley has the good fortune of having three team members who are native Chinese speakers. We were impressed with the performance of University of Central Florida (UCF) on Spanish retrieval at TREC-4, a performance attributed to the availability of native Spanish speakers and students from Mexico who were intimately familiar with the socio-political landscape of the Mexico. The Spanish document collection of TREC-4 came from a Monterrey Mexico news service. Moreover, the UCF group spent an average of 40 hours per topic constructing an elaborate knowledge base for each query, in this way retrieving considerably more relevant documents than were retrieved by the other Spanish participants.

While the Berkeley group did not have the luxury of an extra 1,000 hours for query construction, the Chinese speakers and readers from our team spent about three hours per topic constructing manual queries for Chinese. This effort seems to have paid off with an overall precision of 0.4610 and 10 best-of-topic for our manual run BrklyCH2. BrklyCH2 only missed 35 relevant documents out of 1399 total relevant found for the 19 queries evaluated by the time of the conference. Since BrklyCH2 used negative terms, we also did an unofficial run without negative terms which found 8 more relevant documents and had an average precision of 0.4673.

5.1. Chinese segmentation software

It is well known that the Chinese character set consists of representations similar to phonemes in English, where words are usually composed of up to three adjacent characters. Unlike English and most European languages where word boundaries are distinguished by blank space, Chinese word formations must be distinguished by contextual recognition.

There are two main approaches to find wording boundaries in Chinese (He 1988, Chen and Lieu 1992, Wu and Tseng 1995). The first one uses a set of rules and a dictionary that includes components of one and two Chinese characters. The components in the dictionary are classified into different categories. After the components in the dictionary are recognized in Chinese context, the rules are applied to concatenate the components into a word. The second approach uses an exhaustive dictionary to match the longest string in the context. Since not all the words can be included in the dictionary the remaining strings are divided mechanically into short strings. On the other hand, Chinese has other features which lessen the work of preparing recognition software. In particular verb variants are non-existent, in that tense recognition is performed by the additional words "past" or "future" explicitly included within the text.

For TREC-5, the Berkeley group obtained a public domain Chinese dictionary of 91,000 words from the World Wide Web Chinese software site (<http://www.ifcss.org/ftp-pub/software/data/>). In addition a stop word list of 444 words was constructed manually. The Berkeley group then used its segmentation software to match substrings of Chinese character streams within the TREC-5 document collection against the initial dictionary. Character strings which did not automatically match dictionary words were output and examined manually, and those which were actual Chinese words were added to the dictionary. This process was iterated several times until an additional 46,659 words were added to the Chinese dictionary.

Our segmentation algorithm employed this basic strategy: starting with the first character in a document, the text was matched against a dictionary at each character in sequence. The text is searched from the first character, one character at a time. The longest match found from each starting character is kept, if its last character extends beyond the end of all previous matches.

Any non-matched space was considered unknown. This could be optionally output in a number of ways:

- 1) as single characters;
- 2) as a complete segment;
- 3) both;
- 4) alone, without the matches;
- 5) not output, suppressed.

For TREC-5 we used option 2 and wrote out the complete segment.

6. Chinese results

Our automatic runs, of course, utilized the same segmentation algorithms to construct the queries from the topic descriptions. The official TREC-5 automatic entry BrklyCH1 had overall precision (for, of course, 19 topics) of 0.3192, fourteen percent above the average of median precisions (0.2809) over the fifteen automatic runs. A striking difference between automatic and manual construction can be found in Chinese topic CH14 "Cases of AIDS in China" which uses the common (familiar) word for AIDS used in Hong Kong and Taiwan. This form, which roughly translates as "disease of love," is only found in five documents of the TREC-5 Chinese collection. In other documents the official term for AIDS (which is phonetically similar to the English pronunciation) is used. For the BrklyCH1 automatic run on this topic only 23 out of 57 relevant documents were retrieved (with a precision of 0.0715). For the BrklyCH2 manual run on topic 14 the addition of the official term for AIDS retrieved 46 out of 57 relevant for a precision of 0.4768. Overall the BrklyCH2 precision of 0.4610 was forty-four percent better than the BrklyCH1 automatic run's average precision of 0.3192.

7. TREC-5 Spanish retrieval

The main effort of TREC-5 Spanish work went into improving Berkeley's morphologic stemming software. A new, larger ortho-irregular verb list was obtained from D. German (German 1996), which was split, processed and analyzed. It was then refined to improve upon ambiguous definitions and added to the new Spanish stemmer. For the new Spanish collection, 184,469 verb instances were reduced to 3375 unique verb stems. This stemming seems to have improved our results (there was no change in the ad-hoc retrieval Spanish formula from TREC-4).

In addition, the TREC 5 Spanish queries and collection were modified in some new ways. As acronyms are generally short words that may lose their distinction through stemming and conversion to lower-case, a system of tagging acronyms was developed to try to preserve their uniqueness. This was used automatically on the collection and queries. Secondly, a system of correcting spelling mistakes, especially missing accent marks, was developed using a look-up table generated from a massive, unstemmed wordlist. This most likely made a significant difference in our automatic performance since the queries were full of spelling errors, especially missing accent marks, many of which

our software caught. Finally, our automatic query was expanded such that terms estimated to be important words or names were repeated four times.

The BrklySP5 automatic run using the short descriptive Spanish query had an overall precision of 0.2526 with two runs (Spanish query 57 at precision 0.6726 and Spanish query 68 at 0.5778 precision) achieving best overall performance. The BrklySP6 run was our manual run with negative terms. Its overall precision of 0.3488 was thirty eight percent better than the automatic run.

As an example of the use of negative terms, Spanish topic SP58 on the narcodollar financing of Colombian President Ernesto Samper's election would retrieve documents about Samper visiting disaster areas after earthquakes and volcanic eruptions. For this query we added the negative terms 'avalanch' and 'seismo'. Since such a query also retrieves documents about narcodollars in Mexico and Brazil we added these words to the negative list. The result is that the precision for query SP58 decreased from 0.6421 for BrklySP5 to 0.2075 for BrklySP6. Once again, negative terms have yet to prove their viability.

Time and again it was clear that the addition of native Spanish speakers would have helped in manual query construction. We will be exploring a partnership with the UC Berkeley Center for Latin American Studies for future TREC conferences.

8. Summary

UC Berkeley's participation in TREC-5 led us to a number of different experiments for the routing problem, experiments which were informative but not always successful. Our final approach was to combine evidence from massive query expansion with a regression which weighted the fifteen most important terms of the expansion according to their predictive capacity. Our ad-hoc and foreign language retrieval experiments have proven the robustness of the TREC-2 algorithm which relies on "optimized relative frequencies" as clues. The Chinese experiments show that careful query construction is the fundamental cornerstone of excellent retrieval results.

9. Acknowledgments

Many of the central ideas were originally developed with Professor William Cooper, leader of the Berkeley TREC team for TREC-1 through TREC-3. We continue to use hacked over versions of the SMART system for our retrieval. We are again grateful to Daniel German for his Spanish morphological dictionary. A portion of this work was supported by grant NSF IRI-9630765 from the Database and Expert Systems program of the Computer and Information Science and Engineering Directorate of the National Science Foundation.

10. References

- Chen K Liu S-H (1992) "Word Identification for Mandarin Chinese Sentences," *Proceedings of COLING-92, The 15th International Conference on Computational Linguistics*, Nantes, France, August 23-28, 1992, pp 101-107.
- German, D (1996) private communication. He can be reached at <http://csgwww.uwaterloo.ca/~dmg/home.html>.
- Gey F Chen A (1996) "Intelligent Boolean Filtering for Routing Retrieval," UC DATA Technical Report IS96-1, January 1996, available from the authors.
- Gey F Chen A He J Meggs J (1995) "Logistic Regression at TREC-4: Probabilistic Retrieval from Full-text Document Collections," in (Harman 1995b).
- Harman D, ed. (1995a) *Proceedings of the Third NIST Text Retrieval Conference (TREC3)*, National Institute for Standards and Technology, Washington, DC, November 2-4, 1994, NIST Special Publication 500-225, April 1995.

Harman D, ed. (1995b) *Proceedings of TREC-4, the Fourth Text REtrieval Conference*, National Institute of Standards and Technology, Gaithersburg, MD Nov 1-3, 1995.

Jianzhang He (1987). "The approach and experiments of the automatic word extraction in Chinese Science & Technology documents," *Ching Pao Ko Hsueh (Information Science)*, v.8, no. 4, August 1987, pp 35-45. (in Chinese)

Kwok K (1995), "A network approach to probabilistic information systems," *ACM Transactions on Information Systems*, v. 13, no. 3, July 1995, pp 324-353.

Kwok K (1996) "A New Method of Weighting Query Terms for Ad-hoc Retrieval," *Proceedings of SIGIR96, the 19th Annual International Conference on Research and Development in Information Retrieval*, Zurich, Switzerland, August 18-22, 1996, pp. 187-195.

Robertson S Walker S Jones s Hancock-Beaulieu M and Gatford M, "Okapi at TREC-3," in (Harman 1995a).

Robertson S Walker S, "Some Simple Effective Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval," *Proceedings of SIGIR94, the 17th Annual International Conference on Research and Development in Information Retrieval*, Dublin, Ireland, July 3-6, 1994, pp. 232-241.

Turtle H and W Croft (1991). "Evaluation of an Inference-Network-Based Retrieval Model, *ACM Transactions on Information Systems*, v. 9, no. 3, July 1991, pp 187-222.

Wu Z Tseng G (1995) "ACTS: An Automatic Chinese Text Segmentation System for Full Text Retrieval," *Journal of the American Society for Information Science*, v.46, no. 2, January 1995, pp 83-96.

Berkeley Chinese Information Retrieval at TREC-5: Technical Report

Jianzhang He, Jack Xu, Aitao Chen, Jason Meggs and Fredric C. Gey

UC DATA

University of California

Berkeley, CA 94720

December 1996

1 Introduction

For Chinese track in TREC5, the collection from the People's Daily and Xinhua News articles includes 164,761 documents and the volume of the collection is about 170 MB. There are 28 queries. The task of Chinese track is to submit 1000 documents for each query ranked in the order of likelihood of relevance to the query.

It is a well known problem that there is no separator between Chinese words so that Chinese words cannot be directly used to index or search text as it is allowed in English. Therefore, people used characters, n-grams, or words as search-able tokens (character is 1-gram). In TREC5, we tried to use any meaningful string within text as indexing or search tokens. Our basic strategy is to use an exhaustive dictionary to segment document collection and queries and to use Berkeley TREC2 ad hoc retrieval algorithm to rank retrieved documents.

2 The dictionary

Our method depends heavily on the exhaustive Chinese dictionary. We obtained a public domain Chinese dictionary of 91,000 words and phrases from the World Wide Web Chinese site <http://www.ifcss.org/ftp-pub/software/data>. We expanded our dictionary by working on the TREC5 Chinese collection. The process of the expansion of the dictionary is as the following:

1. segment the collection with the available dictionary.
2. collect unmatched strings.
3. rank the unmatched strings by their frequencies.
4. manually select the top ranked entries.
5. add the selected entries to the dictionary.

The above process was iterated several times until another 46,659 entries were added to the dictionary. The criteria to select an entry is whether the entry is meaningful in Chinese. Therefore, the entries added to the dictionary includes words, idioms, personal names, geographical names, etc.

We also manually constructed a Chinese stopword list which includes 444 entries for which some of them are single character words.

3 The segmentation method

There are different requirements on Chinese segmentation for different applications. For example, machine translation and natural language processing require to segment Chinese text correctly according Chinese syntax. For the purpose of information retrieval, we may not have to segment Chinese text correctly since the ultimate goal is to improve retrieval performance. Although people have used mechanical generated tokens, such as single characters and n-gram, as indexing and search tokens, we

believe that using meaningful tokens is better than using mechanically generated tokens. Therefore, we applied our dictionary as a primary tool to segment the collection and queries.

Segmentation based on a big dictionary can have the following three basic formats. The first is the longest match, for which text is sequentially scanned to match the dictionary. The longest matched strings are taken as indexing and search tokens and shorter tokens within it are discarded. Since longer tokens in the dictionary are more specific, longest match will generate less tokens with more specific meaning.

The second is the shortest match, for which text is scanned sequentially to match the dictionary. The first matched tokens are taken and the match process started from the next character. With the shortest match method, the segmentation process will generate more tokens with less specific meaning.

For example, from the string 数字电子计算机 (digital computer), the longest match will produce the whole string as a token but if we apply the shortest match, we will get the following tokens: 数字 (digit), 电子 (electron), 计算 (compute), and 机 (machine).

The third is the overlap match, for which tokens generated from the text can overlap each other across the matching boundary. For example, the string from query 14, 艾滋病毒 (AIDS virus) is segmented into 艾滋病 (AIDS) and 毒 (poison, toxin) with the longest match method. It is segmented into 艾滋病 (AIDS) and 病毒 (virus) with overlap match. In this case, overlap match produce better result than the regular longest match but we expect it will also generate some negative effects sometimes.

Based on the three basic match methods, we can use some combination of them. For example, we can have longest match plus overlap match, we also can have shortest match plus overlap match, or we can have all three together. For TREC-5, we used longest match plus overlap match. Since no dictionary is expected to include all the words, there are must be some unmatched strings left with dictionary based segmentation. Unmatched strings can be used as tokens for indexing and search. It is also can be used to expand the dictionary. There are several ways to treat unmatched strings:

1. take each character in the string as a token;
2. take the whole string as a token;
3. both 1) and 2);
4. discard.

We used 2) in our TREC-5 runs.

4 Manual query expansions

It is allowed to exam initial search result and then to adjust queries to achieve better performance for any manual ad hoc runs in TREC-5. We reconstructed the 28 queries by checking on our initial search results. We did three things on the queries:

1. add new words;
2. change weights (frequency) of words;
3. add negative words.

The process iterated several time before we obtained our final version of our queries. For example, our manually constructed query 14 is as the following:

Averagely, we spend about 2.8 hours per query to reconstruct the 28 queries. The retrieval result of manual queries improved 40% over our automatic run.

5 Negative terms

In some of the TREC queries, it is indicated that documents in some categories are not relevant to a particular query and sometimes a query will specify that documents in a category are relevant and otherwise are not relevant. For example, a query may only needs to retrieve documents about U.S.A and at the same time documents about other countries are not relevant. We developed the strategy to use negative terms to adjust the final weights of a document to a query. We only apply negative term weight adjust to our manual runs.

Table 1: Query 14: Cases of AIDS in China.

我国 (our country)(16)	艾滋病 (AIDS) (15)	中国 (China)(11)
监测 (control)(6)	发现 (found) (6)	感染 (infect)(6)
病毒 (virus)(6)	预防 (prevent) (5)	H I V (4)
患者 (patient)(4)	防治 (prevent and cure)(4)	
预防为主(prevention first) (3)	病例 (medical cases)(3)	
云南 (Yunnan)(3)	病人 (patient) (2)	
性病 (sexual transmitted disease)(1)	吸毒 (drug use)(1)	
注射器 (syringe)(1)	传染 (infect)(1)	

Table 2: Negative terms for query 14.

美国 (The U.S.)(2)	德国 (Germany) (2)	脊髓 (spinal cord)(2)
灰质 (grey matter)(2)	埃博拉 (2)	脑炎 (encephalitis)(2)
血吸虫 (blood fluke)(2)	丝虫 (filaria) (2)	非洲 (African)(2)
癌症 (cancer)(2)	香港 (Hong Kong)(1)	加拿大 (Canada)(1)
欧美 (The West)(1)		

The adjustment is to reduce the weight of a document if negative terms appear in the document. What we did is to divide the final weight of the document by the square root of 1 plus the sum of the negative term frequency within the document.

Our negative term experiment shows that the current strategy improves precision on the higher rank and degrades on the lower rank. The overall performance is not as good as without negative terms but all the change may be not statistically significant. We still believe that negative terms can play some role but it deserves further investigation.

6 Results

By the time of TREC-5 conference, only 19 out of 28 queries were judged. Our manual run is the best with a 46.10% average precision. We retrieved 1364 out of 1399 relevant documents. we have 10 best and 2 almost best queries out of the 19 queries. We achieved a 97.50% overall recall. Our automatic run averaged 31.92% precision with 15 queries above average. Our average precision is 13.63% better than the median average precision. The following is the detailed list prepared by Fred.

TREC-5 Chinese Manual Summary

10/28/96

Statistics computed over 5 manual runs.

Topic	Relevant Retr. @ 100				Relevant Retr @ 1000				Average Precision		
	#	Best	Median	Worst	Best	Median	Worst	Best	Median	Worst	BrklyCH2
1	13	11	9	7	13	13	11	0.1781	0.1350	0.0489	0.1350
2	69	38	29	27	69	63	58	0.4084	0.2993	0.2377	0.4084*
3	29	26	24	21	29	29	27	0.5791	0.3684	0.2897	0.3684
4	51	27	26	24	51	49	45	0.3130	0.2531	0.2344	0.3127+
5	28	11	9	6	28	25	22	0.0984	0.0732	0.0453	0.0942+
6	77	50	15	1	76	45	32	0.4632	0.0713	0.0144	0.4632*
7	17	12	10	6	17	14	12	0.5029	0.3345	0.2305	0.3345
8	43	41	36	15	43	42	37	0.7889	0.6032	0.1600	0.7889*
9	122	76	72	47	122	117	96	0.7689	0.6950	0.4175	0.7689*
10	49	28	22	14	43	33	29	0.2324	0.1800	0.1154	0.2324*
11	186	72	48	34	182	155	148	0.5472	0.3128	0.2299	0.5472*
12	119	47	39	28	116	113	63	0.3512	0.3100	0.1557	0.3512*
13	110	61	40	31	107	107	68	0.4875	0.3371	0.2089	0.4875*
14	57	50	34	21	56	46	38	0.6074	0.4768	0.1590	0.4768
15	71	50	41	6	71	64	33	0.5939	0.4535	0.0472	0.5939*
16	59	30	26	22	56	56	42	0.3602	0.3008	0.1519	0.3008

21	238	95	91	88	237	231	207	0.8956	0.8035	0.6138	0.8956*
22	15	15	15	11	15	15	15	0.7690	0.7481	0.2409	0.7481
23	46	30	25	18	46	45	40	0.5222	0.4521	0.2678	0.4521
1399 total relevant found								Average over 19 queries		0.3794	0.4610
1364 found by BrklyCH2 at 1000											

* = best

+ = almost best

TREC-5 Chinese Automatic Runs with comparison to BrklyCH1

10/28/96

Statistics computed over 15 automatic runs.

Topic	# Rel.	Relevant Retr. @ 100			Relevant Retr @ 1000			Average Precision			BrklyCH1
		Best	Median	Worst	Best	Median	Worst	Best	Median	Worst	
1	13	13	9	0	13	13	0	0.1972	0.1021	0.0000	0.1021
2	69	41	29	11	68	60	23	0.4365	0.2390	0.0244	0.3079
3	29	26	23	6	29	28	10	0.5228	0.3482	0.0178	0.4144
4	51	29	24	6	51	50	18	0.3690	0.2442	0.0247	0.3683+
5	28	13	9	6	28	22	8	0.1090	0.0612	0.0374	0.0471-
6	77	24	12	2	74	45	25	0.2116	0.0640	0.0163	0.2006+
7	17	13	9	5	16	13	8	0.4613	0.2896	0.0612	0.2096-
8	43	31	18	6	43	37	20	0.4462	0.1717	0.0264	0.3376
9	122	76	58	9	122	116	18	0.7575	0.5355	0.0117	0.3485-
10	49	26	14	3	45	32	11	0.2899	0.1293	0.0102	0.1700
11	186	47	39	15	179	155	49	0.4069	0.2643	0.0275	0.3059
12	119	40	30	17	119	105	41	0.3381	0.2348	0.0691	0.2165-
13	110	45	27	3	108	101	3	0.3767	0.1819	0.0016	0.1818
14	57	22	8	2	57	21	2	0.2562	0.0548	0.0008	0.0715
15	71	59	36	2	71	65	16	0.7172	0.4507	0.0222	0.5141
16	59	37	21	2	59	56	10	0.4506	0.2915	0.0046	0.2915
21	238	96	90	16	237	230	40	0.9083	0.7956	0.0248	0.7956
22	15	15	13	4	15	15	4	0.7754	0.4914	0.0667	0.5346
23	46	40	26	0	46	44	0	0.6658	0.3874	0.0000	0.6473
1399								Average precision over 19 queries		0.2809	0.3192
1246 relevant found by BrklyCH1											

- = below median

+ = close to best

After TREC-5 conference, the 28 queries were all judged. Our average precision is 49.05% and we retrieved 2095 documents out of 2182 relevant documents. Our overall recall is 96.01%. We believe our manual run is still ranked first. Our automatic run averaged 35.68% precision and retrieved 1948 documents out of 2182 relevant documents with the overall recall of 89.28%.

7 Credits

Dr. Fred Gey provided the leadership and methodology advisory for the project. Jianzhang He is the coordinator of the Chinese track project. The following is the list of people and their contribution for Berkeley TREC-5 Chinese retrieval project.

1. Overall methodology development: Jianzhang He, Fred Gey.
2. Manual dictionary expansion: Jianzhang He, Jack Xu.
3. Stopword list construction: Jack Xu, Jianzhang He.
4. Chinese collection filtering: Jack Xu.
5. Segmentation software coding: Jason Meggs.
6. Retrieval software: Aitao Chen.
7. Interactive document check software: Aitao Chen.
8. Modifying SMART to handle Chinese text: Jianzhang He.

9. Negative term retrieval software implementation: Jianzhang He.

10. Manual query reconstruction: Jianzhang He, Jack Xu, Aitao Chen.

Prof. Bill Cooper also provided helpful advises during the process of the project. Berkeley TREC-5 Chinese Retrieval result was presented by Jianzhang He at NIST on 11/21/96.

TREC-5 Experiments at Dublin City University: Query Space Reduction, Spanish & Character Shape Encoding.

Fergus Kelleedy & Alan F. Smeaton

{fkelleedy, asmeaton}@compapp.dcu.ie

School of Computer Applications,
Dublin City University, Glasnevin,
Dublin 9, IRELAND

Abstract

In this paper we describe work done as part of the TREC-5 benchmarking exercise by a team from Dublin City University. In TREC-5 we had three activities as follows:

- Our ad hoc submissions employ Query Space Reduction techniques which attempt to minimise the amount of data processed by an IR search engine during the retrieval process. We submitted four runs for evaluation, two automatic and two manual with one automatic run and one manual run employing our Query Space Reduction techniques. The paper reports our findings in terms of retrieval effectiveness and also in terms of the savings we make in execution time.
- Our submission to the multi-lingual track (Spanish) in TREC-5 involves evaluating the performance of a new stemming algorithm for Spanish developed by Martin Porter. We submitted three runs for evaluation, two automatic, and one manual, involving a manual expansion from retrieved documents.
- Character shape coding (CSC) is a technique for representing scanned text using a much reduced alphabet. It has been developed by Larry Spitz of Daimler Benz as an alternative to full-scale OCR for paper documents. Some of our TREC-5 experiments have started evaluating the performance of a CSC representation of scanned documents for information retrieval and this paper outlines our future work in this area

Acknowledgements: The authors would like to acknowledge the contributions to the work reported here given by Martin Porter who developed and provided us with access to his Spanish stemming algorithm, and to Larry Spitz who developed the character shape coding technique and who also provided us with access to his code and his accumulation of knowledge and data in the area.

1. Introduction

The work reported in this overview is a description of the research we carried out for our efforts in TREC-5. In all we have submitted 13 official runs, DCU961 To DCU964 (4 runs) represent Category A ad hoc, DCU965 to DCU967 (3 runs) represent our work in Spanish and DCU968 to DCU96D (6 runs) represent our efforts in Character Shape Coding. These 13 submitted TREC-5 runs are based on three separate streams of research and are described independently in the following three sections of the paper.

For all our work, the platform we used for our experiments was a SUN SparcStation 5 with 64 Mbytes of RAM and 6 Gbytes of local disk space. All experiments detailed in this overview were carried out using our own IR search engine developed by us over the last two years.

2. Query Space Reduction

In this line of our research the issue of query response time is critical and is the motivational force for the work, as in the case with research carried out by [Bown95], [Pers95] and [Moff94]. It is, in our opinion, very important that IR systems return required information to a user in an acceptable amount of time and for us this is of equal importance as the effectiveness of an IR system being used. Traditional IR research has always concentrated on effectiveness and efficiency has been a poor relation. To this end we developed and implemented a number of Query Space modelling techniques which improve the efficiency of our experimental IR system.

Within our test environment we define the Query Space (QS) to be the amount of data from the inverted file which needs to be processed in order to satisfactorily respond to a users query. Figure 2.1 illustrates an abstract view of this data.

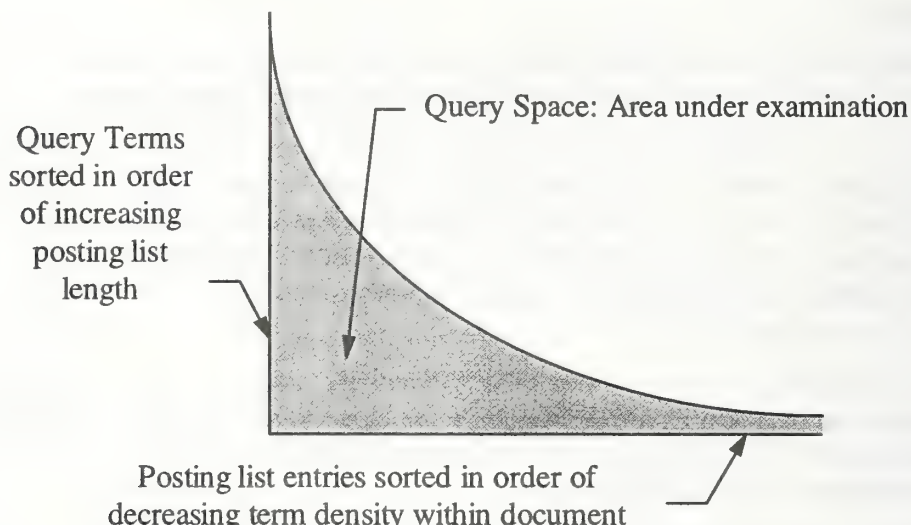


Figure 2.1 - Abstract View of Query Space.

The QS is composed of query terms and their corresponding posting lists. Query terms in the QS are those which occur both in the query and the document collection, i.e. their inclusion in the QS will have some impact on the final ranking of documents returned to the user in response to the query. For visualisation purposes the QS is best laid out in two dimensional space with the y-axis representing the query terms and the x-axis representing the posting lists of the query terms.

Within the QS, query terms are sorted in order of increasing posting list length. The posting lists themselves are ordered by decreasing within document term density. Due to the structure of the QS in which the posting lists are arranged vertically and horizontally in order of probable value to the query, one can assume that the concentration of relevant information with respect to the query is higher in the top left part of Figure 2.1 than in the bottom right part. This QS structure facilitates the notion of thresholding, in which not all of the QS need be processed thus greatly reducing I/O during retrieval.

2.1 Query Term Thresholding.

Query Term Thresholding (QTT) exploits the structure of the QS by removing the necessity to process all terms in the query. Figure 2.2 illustrates QTT as it applies to our QS, the shaded area represents the subsection of the QS that is actually processed during retrieval. This thresholding technique operates at the query term level i.e. based on the threshold value a query term is either included or excluded from the retrieval process. As one moves down the QS,

terms become more general (they occur in more documents) and therefore less useful in discriminating one document from another. These terms also occupy a significant percentage of the QS due to their frequency of occurrence within the collection.

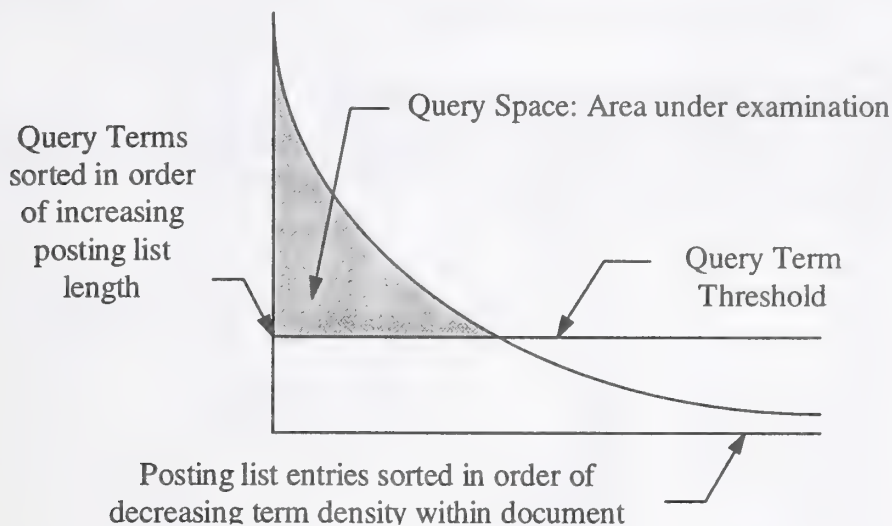


Figure 2.2 - Abstract View of Query Term Thresholding.

It therefore makes sense to attempt to reduce or eliminate the need to process these 'low value' query terms during the retrieval process. The incorporation of the QTT significantly reduces the processing and I/O costs of the retrieval process whilst also having a positive effect on the effectiveness of the retrieval process by eliminating 'noisy' postings from consideration during retrieval.

2.2 Posting List Thresholding.

As the posting list entries are sorted in order of decreasing 'within document term density', posting entries at the end of a posting list will be of less value in the retrieval process due to their low within-document term density. This fact presents the possibility of removing these 'low value' postings from consideration during the retrieval process. It makes sense that that more of the discriminating posting list entries (those with high IDF scores) entries should be processed and less of the non-discriminating posting list entries should be processed.

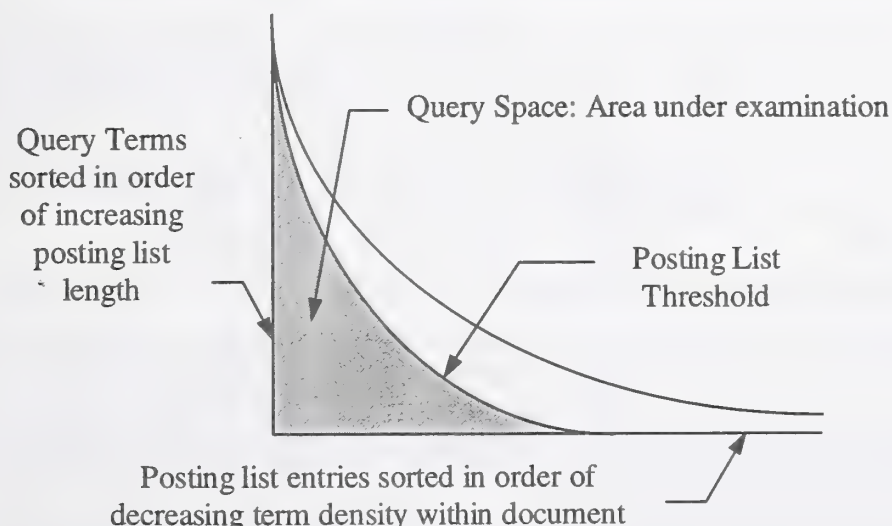


Figure 2.3 - Abstract View of Posting List Thresholding.

This results in a variable thresholding approach in which the PLT value is initially set to a high percentage of postings and is gradually lowered as each QS index term is processed. Figure 2.3 abstractly illustrates this thresholding process.

2.3 Query Term and Posting List Thresholding.

The QS thresholding techniques detailed in Sections 2.1 and 2.2 can operate in conjunction with each other on the same QS as illustrated in Figure 2.4 resulting in further reductions in the amount of the QS processed during retrieval.

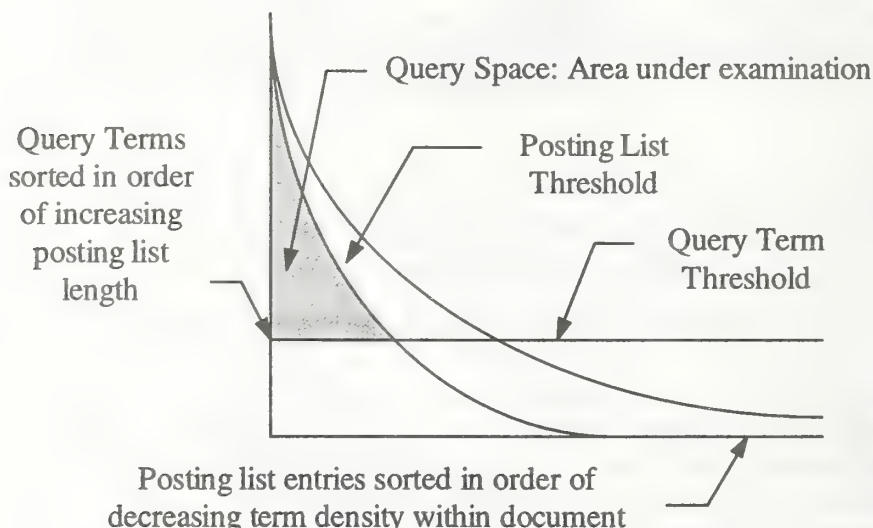


Figure 2.4 - Abstract View of Combined Thresholding Approach.

2.4 Document Accumulator Thresholding.

Document accumulators are used for summing up all of the partial query term similarity scores to form a final query-document similarity score. If no restriction is imposed on the activation of document accumulators then a large number of accumulators must be sorted in order to identify the N most highly-scored documents. Therefore some method for limiting the number of accumulators activated during retrieval is very important for efficiency.

The operation of our accumulator thresholding technique is as follows: new accumulators are created for all documents which achieve a non-zero query document similarity score until the maximum active accumulator limit is reached. Once reached, only the already activated accumulators or registers are allowed sum more partial query document similarity scores. This places an upper limit on the number of accumulators activated hence reducing the load on the sort procedure which produces a final ranked list of documents.

2.5 Results of Query Space Reduction.

Figure 2.5 and Figure 2.6 detail the results obtained in terms of system effectiveness for our ad hoc submissions to TREC-5. We submitted two automatic runs (*DCU962*, *DCU964*) and two manual runs (*DCU961*, *DCU963*). *DCU961* and *DCU962* have the QSR techniques outlined earlier applied to them while *DCU963* and *DCU964* have no thresholding. The parameters chosen for these respective thresholding methods have been defined through experiments on the TREC-4 and TREC-3 collections. It can be clearly seen that there is no performance degradation in terms of system effectiveness caused by the inclusion of our QSR techniques. In fact the inclusion of our QSR techniques actually causes a slight improvement in

performance in terms of average precision for both the automatic (Avg. P moving from 0.1334 to 0.1340) and manual runs (Avg. P moving from 0.1804 to 0.1862).

AUTOMATIC			MANUAL		
	DCU962 (QSR)	DCU964 (No QSR)		DCU961 (QSR)	DCU963 (No QSR)
P @ 0.0	0.4525	0.4404	P @ 0.0	0.5336	0.4952
P @ 0.1	0.2540	0.2531	P @ 0.1	0.3615	0.3507
P @ 0.2	0.2076	0.2067	P @ 0.2	0.2979	0.2897
P @ 0.3	0.1805	0.1802	P @ 0.3	0.2571	0.2492
P @ 0.4	0.1575	0.1571	P @ 0.4	0.2254	0.2171
P @ 0.5	0.1374	0.1380	P @ 0.5	0.1955	0.1883
P @ 0.6	0.1079	0.1085	P @ 0.6	0.1478	0.1437
P @ 0.7	0.0878	0.0843	P @ 0.7	0.1173	0.1136
P @ 0.8	0.0644	0.0639	P @ 0.8	0.0814	0.0756
P @ 0.9	0.0277	0.0275	P @ 0.9	0.0403	0.0367
P @ 1.0	0.0181	0.0192	P @ 1.0	0.0181	0.0177
Av. P	0.1334	0.1340	Av. P	0.1862	0.1804
P @ 10 docs	0.2540	0.2460	P @ 10 docs	0.3320	0.3160
P @ 30 docs	0.1867	0.1880	P @ 30 docs	0.2547	0.2427

Figure 2.5 - QSR Vs No QSR for Automatic and Manual runs.

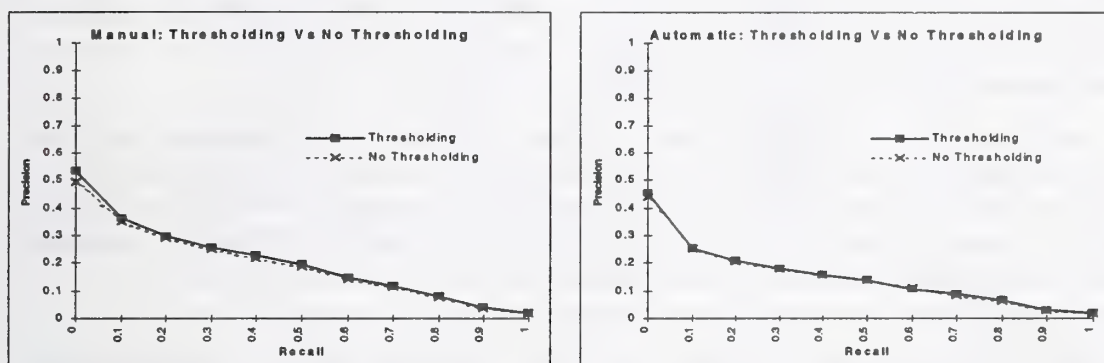


Figure 2.6 - Precision-Recall comparisons for Automatic & Manual runs.

In addition to measuring the effect of our QSR techniques on system effectiveness we also measured the effect on system efficiency in terms of CPU resources used during retrieval. For all runs the memory and processor loads were the same given the machine used was dedicated to this task.

	Automatic			Manual		
	Average	Max	Min	Average	Max	Min
No QSR	9.018	16.9	2.1	22.442	44.4	9.8
QSR	4.644	8.6	2.1	8.238	19.4	4.5
% Red.	48.50%			63.29%		

Figure 2.7 - Timing comparisons (in Seconds) for Automatic & Manual runs.

Figure 2.7 summarises the results we obtained for all of our ad hoc TREC-5 submissions. The average time spent processing a query for the automatic submission goes from 9.018 seconds to 4.644 seconds which represents a saving of 48.50% and the average time spent processing a query for the manual submission goes from 22.442 seconds to 8.238 seconds a

saving of 63.29%. The difference in savings obtained between the automatic and manual runs can be ascribed to the difference in the average size of the Query Spaces generated, with our system identifying on average 8.96 terms per query for the automatic run and an average of 30.9 terms per query for the manual run. The larger the QS generated the greater potential savings possible using our QSR techniques.

3. Spanish TREC-5

In TREC-3 and TREC-4, Dublin City University participated in the Spanish track. In TREC-3 we implemented a trigram matching procedure where trigrams were assigned IDF weights. In TREC-4 we used the NMSU Spanish POS tagger to index documents and texts by the base forms and POS tags for words and again used term weighting for retrieval.

Our involvement in the Spanish track for TREC-5 came out of a chance conversation with Martin Porter, developer of the much-used 1980 stemming algorithm for English. He now works for a company called MUSCAT who have built up a reputation for their products in the museum catalogue area and from that have developed a search engine for WWW sites called *Euroferret*. *Euroferret* is used by the Electronic Telegraph online newspaper and by UK government departments for their respective web pages. In an effort to expand the capabilities of *Euroferret*, Martin Porter has developed a stemming algorithm for Spanish (as he has done for Italian and for French). The issue of Muscat LTD's intellectual property rights prevent a fuller description of the Spanish stemming algorithm and the details of its operation have not been published, but we include here a sketch of how it operates.

The basic plan for the Spanish stemmer is like that of the English stemming algorithm. Word endings are taken off if they come at the end of a stem of suitable length, as measured in syllables. As an ending is removed, the stemmer may attempt to remove further endings associated with it. Thus the ending -IVO or -IVA (like the English -IVE) has the associated ending -AT, and after taking off -IVO the stemmer looks for -AT, possibly taking off -ATIVO. There is a small graph structure for these endings which the implementation follows.

Porter's Spanish stemmer also has a short list of prefixes, which do not contribute to the evaluated length of the stem, for example CON-. If there is no significant morphological suffix of this type, the stemmer takes off a verb ending. This is one of the things that makes Porter's Spanish stemmer so different from and bigger than his stemmer for English. The romance languages all have highly inflected verb forms: seven tenses, each with six endings as well as imperative and participle forms, each having three different classes. Spanish and Italian also have forms in which direct and indirect object pronouns attach to the verb. All of these features are handled by the stemmer.

The stemmer takes special actions for irregular verbs. The English language has about 100 irregular verbs in use, e.g. DIG, DRAW, DRINK, DROVE, etc., but the irregularities are very simple (DIG/DUG, DRAW/DREW/DRAWN, etc.) and the irregular forms of verbs frequently mean different things, so any kind of normalisation is unwise (DUG can also mean a teat, DRUNK can also mean inebriated, DROVE can mean a large number of, etc.). In Spanish there are about 500 irregular verbs that Porter's stemmer knows about and their complexity suggests normalisation is advisable. To this list of irregular verbs one can add a batch of regular verbs with very short stems. The stemming algorithm then searches the whole word for a match with one of these verbs in one of its declinable forms and if found, can normalise it exactly.

For our involvement in TREC-5 Spanish we created our own list of Spanish stopwords, using a translation of Porter's stopword list for English supplemented with additional words from a Spanish grammar book for prepositions and other closed word categories. We removed

stopwords and stemmed the document and query texts in the TREC-5 collection. Using the search engine and term weighting functions developed for our mainline ad hoc experiments described earlier, we completed 3 official runs as follows:

- **DCU966**: we used query terms from the short form of the queries only (automatic); the average query length was 16.0 terms
- **DCU965**: we used the full form of the query only (manual); the average query length was 48.5 terms.
- **DCU967**: we used the short form of the query (run DCU966) to select the top 10 documents which were then presented to a Spanish speaker to read. Using these documents the user then selected extra terms thus constructing an expanded form of the query which, along with the original query terms, is used to re-rank the remainder of the unseen documents yielding an average query length of 59.2 terms.¹

Our results are reproduced in the table below.

	DCU966	DCU965	DCU967	DCU967C
P @ 0.0	0.6905	0.7661	0.8565	0.7149
P @ 0.1	0.6268	0.6335	0.6428	0.6171
P @ 0.2	0.4985	0.5383	0.5547	0.5147
P @ 0.3	0.4227	0.4719	0.4733	0.4629
P @ 0.4	0.3700	0.3985	0.4213	0.3976
P @ 0.5	0.3299	0.3404	0.3533	0.3327
P @ 0.6	0.2773	0.2819	0.2660	0.2630
P @ 0.7	0.2217	0.2213	0.1954	0.1935
P @ 0.8	0.1744	0.1697	0.1468	0.1458
P @ 0.9	0.1192	0.1020	0.0927	0.0923
P @ 1.0	0.0026	0.0021	0.0004	0.0004
Av. P	0.3298	0.3482	0.3482	0.3258
P @ 10 docs	0.4960	0.5760	0.5760	0.5000
P @ 30 docs	0.4507	0.4813	0.4820	0.4307

As would be expected, using the longer form of the queries (DCU965) yields better performance than using the shorter form (DCU966). Manually expanding the query from the initial short form (DCU967C) improves retrieval over simply using the short form at the high precision end of the scale, but not overall. Our manual expansion of queries by a Spanish speaker (native Irish) with no information retrieval background (DCU967C) does not yield as good retrieval as using the long form of the queries (DCU965).

In terms of comparison to other submitted results, the table below shows how our overall average precision compares with others' submitted runs.

	DCU966	DCU965	DCU967C
On or above Median	11	17	14
Below Median	14	8	11

¹ In our official submissions we erroneously submitted results for the complete collection re-ranking after query expansion and did not incorporate the top 10 rank positions of our overall ranking having been generated from the first pass. This has been corrected in the results coded as DCU967C and as the reader can see this has a big effect on results.

4. Character Shape Coding

Character shape codes (CSCs) are a reduced alphabet for representing the textual content of documents. In conventional optical character recognition, each token in a scanned image is mapped to a character in the full alphabet consisting of {A-Z, a-z, 0-9}. CSCs, developed by Larry Spitz, are an attempt to define a much-reduced alphabet based on the characteristics of character recognition from bitmaps of scanned documents. In essence CSCs are an attempt to group together characters based on the similarities of their printed characteristics. For example, in most printed fonts the letters "c" and "e" have similar characteristics in that neither rise above a horizontal line defined by the upper reaches of the letter "x", neither drop below the base line whereas the characters g, q, p, etc. do, and both "c" and "e" have an "eastward" concavity.

In previous work by Larry Spitz [Spit95a, Spit95b, Reyn95] he has defined a number of sets of CSCs using different alphabet mappings. A series of evaluations have also been carried out to examine the uniqueness of the CSC representations for the surface forms of words as they occur in text and these have illustrated a surprising uniqueness among CSC patterns. In our CSC experiments in TREC-5 we transformed the representation of document texts into 3 different CSC representations using the 3 different CSC mappings shown below.

CSC-V0

A	A-Z b d f h k l t
x	a c e m n o r s u v w x z
g	g p q y
i	I
j	j

CSC-V1

A	A-Z b d f h k l t
x	a m n o r s u v w x z
e	c e
g	g p q y
i	I
j	j

CSC-V2

A	A-Z b d f h k l t
x	a m o r s u v w x z
e	c e
g	g p q y
i	I
j	j
n	n

In effect what we are simulating is scanning and automatic transformation of paper documents into their CSC representations which is a much simpler and more accurate process than full-scale optical character representation and in our experiments we assume 100% accuracy in such CSC recognition. In later experiments we plan to introduce the naturally-occurring noise which such a process would have.

The effectiveness of a CSC representation for documents for information retrieval applications has not been evaluated before so our CSC experiments in TREC-5 we set out to do some preliminary work in this, using TREC to establish the groundwork for future experiments. In order to concentrate our work on the effectiveness of the CSC representations, we chose to do our experiments in category B ad hoc retrieval, using the 253 Mbyte text (74,520 documents) from the Wall Street Journal. In doing this we eliminated considerations of document length normalisation which have such a variance on experimental results as other

TREC groups have shown. The downside of doing this, however, is that we exclude ourselves from direct comparison with other groups taking part in the corruption track in TREC.

Having “indexed” documents by representing them as the CSC representations for all their surface word forms, we then processed queries in the following way. For both the manual (short form) and automatic (long form) queries we automatically removed stopwords and reverse-stemmed the non-stopwords. This was done by using Porter’s stemming algorithm to stem a large portion (50 Mbytes) of WSJ texts and recording, for each unique word stem, the surface word form occurrence from the WSJ texts, of the word which yielded that stem. This pre-processing stage was used to generate our mapping of word stems to word occurrences. Each word stem in the query was then expanded into the set of surface forms of words in order to generate morphological (due to verb tense, noun plurals, etc.) and surface (due to various letter capitalisations) variants. The set of expanded terms per query was then pruned manually to eliminate errors due to stemming and unlikely word form occurrences. The remaining word form occurrences were turned into each of the three CSC representations and used as query terms for the three sets of experiments.

As an example, query 251 is :

“Exportation of Industry

Documents will report the exportation of some part of U.S. Industry to another country.

Relevant documents will identify the type of industry being exported, the country to which it is exported; and as well will reveal the number of jobs lost as a result of that exportation.”

If we examine the term “*exportation*” as used in the query, it is stemmed to “*export*” which is the same stem as the following word forms which occur in our WSJ text *Export, export, Exports, Exports, exporting, exported*, etc. After manual pruning of the automatic generation of all possible word forms, the following is the list of query terms that was generated for q251:

type	types	lost	Export	Exporting	export	exported
exporter	exporters	country	countries	industry	industrial	industrialised
Industry	job	jobs	result	resulted	resulting	resultant

Each of these surface form occurrences was turned into a CSC token for each of the three mappings we use and the process described above yielded an average of 24.96 terms (CSC “words”) for the short form of queries, and 29.4 terms for the long form of queries.

Each of these CSC terms in each mapping will map to multiple surface form occurrences in the document texts, and the distribution of these is very skewed. For example, the CSC-V1 code “*exxgxxA*” in query 251 above can map to the words *copout, expand* and *export* and in a comprehensive lexicon gathered by Lary Spitz these three surface word forms have been found. In our WSJ texts only the latter two are present. The CSC-V1 code “*ixAxxAxixA*” maps to the word *industrial* only both in Spitz’s comprehensive lexicon and in our WSJ text. At the other extreme, the CSC-V1 code for the word *lost* is “*AxxA*” and shares that code with 194 other word forms in our WSJ text and 366 other words in Spitz’s lexicon ! The table below shows, over the 50 short and long form queries in TREC-5, the average number of surface form occurrences which share the same CSC string, for each of the three CSC mappings we used, as measured in our WSJ text. It is to be expected that these numbers would be higher if

we had not used a domain-specific lexicon (our 50 Mbytes WSJ text) but a comprehensive lexicon instead.

	Short form of queries:	Long form of queries:
CSC-V0	24.88	23.72
CSC-V1	21.37	16.87
CSC-V2	13.30	10.41

This table confirms the expected results of the higher CSC mappings to word forms being more precise and less ambiguous and also shows a high degree of ambiguity with respect to surface form word occurrence, for the CSC tokens we use in queries. This is much higher than we would have expected

In our official TREC-5 category B submission there were 6 runs coded DCU968, -9, -A, -B, -C, and -D representing the use of short and long forms of queries for the three CSC mappings we used. Results are pretty poor and in the table below we show the total number of relevant documents found (from 1064 in total !) for each run:

Code	Meaning	Total reldocs Retrieved (45 topics)	Total reldocs Retrieved (36 topics)
DCU968	CSC-V0, short queries	21	21
DCU969	CSC-V0, long queries	21	20
DCU96A	CSC-V1, short queries	21	21
DCU96B	CSC-V1, long queries	20	19
DCU96C	CSC-V2, short queries	25	24
DCU96D	CSC-V2, long queries	17	16

The poor results above are easily explained by the noise introduced through the use of CSC tokens in queries which have such a high degree of surface form occurrences in text. Clearly the next set of experiments we should perform should involve manually or automatically selecting CSC terms for inclusion in queries based on the number of surface form occurrences sharing that CSC term. This is the subject of ongoing work..

5. Conclusions

The experiments reported in this paper as the Dublin City University submissions to TREC-5 represent mixed successes. The work on query space thresholding yields major improvements in retrieval efficiency without loss of retrieval effectiveness. Our experiments in Spanish track leave us about mid-table in terms of effectiveness compared to others' submissions though we have not incorporated any "smarts" such as phrase recognition, into the retrieval we used in our Spanish runs. The real question in our Spanish runs would be addressed by stemming the documents and texts with a different stemmer and measuring

retrieval effectiveness using the same retrieval environment (term weighting, etc.) we have used here. The final set of experiments on character shape coding must be acknowledged as being only preliminary work as is explained in the analysis of results and further work on CSC-based retrieval is presently ongoing.

6. References

- [Bown95] "Fast Evaluation of Structured Queries for Information Retrieval", E. W. Brown, in: *Proc. SIGIR'95*, Seattle, 1995.
- [Pers95] "Document Filtering for Fast Ranking", M. Persin, in: *Proc. SIGIR'94*, Dublin, 1994.
- [Moff94] "Fast Ranking in Limited Space", A. Moffat, J. Zobel, in: *Proc. ICDE'94*, Houston, 1994.
- [Reyn95] "Document Reconstruction; A Thousand Words from one Picture", J C Reynar et al, in: *Proc. Symposium on Document Analysis and Information Retrieval*, UNLV, 1995.
- [Spit95a] "Using Character Shape Codes for Word Spotting in Document Images", A. L. Spitz, In *Shape, Structure and Pattern Recognition*, D Dori and A Bruckstein (Eds), World Scientific, Singapore, 1995.
- [Spit95b] "An OCR Based on Character Shape Codes and Lexical Information", A L Spitz, in: *Proc. 3rd International Conference on Document Analysis and Recognition*, Montreal, Canada, 1995.

The MDS Experiments for TREC5

Marcin Kaszkiel

Phil Vines

Ross Wilkinson

Justin Zobel

Department of Computer Science, RMIT
{marcin,phil,ross,jz}@cs.rmit.edu.au

1 Introduction

The Multimedia Database Systems (MDS) group at RMIT is investigating many aspects of information retrieval of relevance to TREC. Current work includes combination of evidence, Asian-language text retrieval, passage retrieval, collection fusion, and efficient retrieval from large collections. Here we report on results from three of these strands of research.

2 Dynamic Passage Retrieval

Much of the research in text retrieval has focussed on retrieval of whole documents. However, there are many contexts in which it is preferable to consider retrieval of subparts of documents, or passages, which are potentially a better mechanism for identification of relevance than ranking of whole documents. In this section we explain our new approach to passage retrieval.

Use of passages to rank documents should be particularly effective for high-precision retrieval—because they are good at identifying documents with highly relevant parts—and for collections of longer documents where, by whole-document ranking, the relevance of a passage may be obscured by low relevance overall. In earlier experiments (in the second TREC in 1993, reported in detail later [2]) we observed exactly this behaviour. We have since revisited these experiments and results are discussed below; in these experiments with non-overlapping passages of approximately 2000 bytes (or roughly 300 words) and queries on FR, recall-precision improves from 0.243 for whole-document ranking to 0.328 with ranking based on passages.

However, our analysis of the fixed-passage approach shows that, although it can result in improvements overall, it is not foolproof. For some queries, the best effectiveness was given by use of short passages, of say 50 words, while for other queries long passages were best. Fixing passage length at database creation time is not necessarily desirable.

We therefore chose to explore a dynamic approach, in which passages are determined by queries; thus for different queries a given document could be divided in different ways. Our aim, in effect, was to explore the potential of passage retrieval and allow any piece of a document to be a passage—and to worry later about practicalities!

Given arbitrary resources this exploration of passage retrieval would proceed by, for each document, extracting the passage of every length starting at every word; then computing the similarity of every such passage to the query; then ranking documents according to their highest-ranked passage. Thus a document of 1000 words would yield approximately 500,000 passages; 10,000 words would yield 50,000,000 passages. We had to concede that this might

be impractical. As an approximation we chose a set of fixed passage lengths—from 50 to 600 words in increments of 50, yielding twelve different lengths—but still allowed passages to start at every word in each document. A document of 1000 words thus yielded about 8,100 passages, 10,000 words about 116,100 passages. For each passage length we kept the highest similarity value for each document, or twelve values per document overall. (These are created on the fly, compared to every query, and discarded. A full run, for 50 queries, takes about one week on a SPARC 20.)

We then ranked documents according to their highest-ranked passages. This is not the only way to evaluate a passage-retrieval mechanism, but it fits the TREC methodology and does evaluate the effectiveness of using passages to retrieval whole documents. Nor is it a full test of our “idealistic” experiment, but it should provide good evidence of whether dynamic passages can lead to performance improvement.

In the experiments passage length was one parameter. Another was similarity measure. We used the cosine measure, as this was most successful in preliminary experiments, defined by:

$$C(q, p) = \frac{\sum_{t \in q \cap p} (w_{q,t} \cdot w_{p,t})}{W_p}$$

where q is the query, p is the passage,

$$\begin{aligned} W_p &= \sqrt{\sum_{t \in p} w_{p,t}^2}, \\ w_{p,t} &= \log(f_{p,t} + 1), \\ w_{q,t} &= \log(f_{q,t} + 1) \cdot \log(N/f_t + 1), \end{aligned}$$

$f_{x,t}$ is the frequency of t in x , there are N documents, and f_t is the number of distinct documents containing t . We call this *C-standard*.

When comparing passages of different length, it is clear that normalisation is problematic. For fixed-length passages, having 150 to 350 words per passage provides the best overall performance. But for the above cosine formulation passages of 50 words almost always had the highest similarity values—evidence that length normalisation is poorly formulated. We therefore investigated other approaches to normalisation. One was pivoted document length normalisation [5]. Formally it is inapplicable (it requires averages over all documents in the collection, which is not meaningful in this context) but it has been argued that the length formulation

$$W_p = (1.0 - S) \cdot L + (S \cdot U_p)$$

is reasonably robust, where $L = 300$ words is a typical passage length, the slope S is 0.2, and U_p is the number of unique terms in p . We call this *C-pivot*.

Another approach to normalisation is to standardise the similarities for each query and passage length. By, for each passage length, dividing each similarity by the highest similarity for that passage length, similarities from different passage lengths can in effect be compared according to their ranking amongst passages of the same length. (In a practical implementation this information would not be available, but should it be successful it would provide a direction in which to search for better solutions to this problem.) We call this *C-scaled*.

The full set of experiments were as follows. For baselines we applied C-standard to full documents and to passages as defined by fixed-length “pages” of 2000 bytes and the TextTiling

	Precision at N documents					Avg. prec.	Avg. (R) prec.
	5	10	20	30	200		
C-standard:							
Documents	0.2222	0.2333	0.1595	0.1540	0.0724	0.2434	0.1775
Pages	0.2619	0.2619	0.2214	0.1905	0.0688	0.3278	0.3183
TextTiles	0.2349	0.2238	0.1929	0.1635	0.0650	0.2692	0.2529
C-pivot	0.2556	0.2238	0.1714	0.1571	0.0667	0.2852	0.2461
350 words	0.2762	0.2714	0.2405	0.2048	0.0748	0.3413	0.3364
Passages:							
C-standard	0.2508	0.2333	0.2024	0.1714	0.0712	0.2953	0.3053
C-pivot	0.2635	0.2524	0.2143	0.1921	0.0743	0.3304	0.3297
C-scaled	0.2587	0.2429	0.2262	0.1936	0.0745	0.3487	0.3554

Table 1: *Experiments with FR, disks 1 and 2, queries 51–100.*

method [1]. We applied C-standard to passages of each of the twelve lengths separately, to identify which passage length worked best. We then applied C-standard, C-pivot, and C-scaled to passages of all lengths together; these final experiments emulate dynamic passage retrieval, since in each case a passage of any length can determine the rank of a document. We did not explore combination of evidence. In all these experiments we used the full queries.

We used three data sets, focussing on FR because we would expect the greatest improvement to come from databases of long documents. The first data set was the FR data from disks 1 and 2 and the 21 queries between 51 and 100 that had at least one relevant document in FR. Results are shown in Table 1. Pages of 2000 bytes are the most effective of the previous methods, considerably improving on C-standard and C-pivot on whole documents. Slightly better was fixed-length passages of 350 words, the most effective performance observed for fixed-length passages. Dynamic passages did not significantly outperform fixed-length passages.

The second data set was the FR data from disks 2 and 4 and the 26 queries between 251 and 300 that had a relevant document in FR—these are the “blind” experiments from this year’s TREC. Results are shown in Table 2. In this case passages of 150 words worked best, markedly outperforming the previous methods, the strongest of which was TextTiles. C-pivot, our submitted run, has worked best. The greatest improvement is seen in the top few documents retrieved, with precision increasing from 0.1513 for C-standard to around 0.19 for the passage methods.

The third data set was the full contents of disks 2 and 4 and queries 251–300, that is, this year’s TREC run. We would not expect passage retrieval to result in large improvements on this data set, since most of the documents are fairly short; and, according to the relevance judgments, a disproportionate number of the judged documents are short (yet more evidence, perhaps, of problems with normalisation). In comparison to other retrieval techniques, we would not expect to perform especially well; C-standard is fairly effective, but we have not applied any refinements—phrase extraction and so on—that might yield increased effectiveness. Such techniques are, we believe, orthogonal to passage retrieval. However, we expected to perform at least as well as *C-standard*. Results are shown in Table 3; passages give a marginal overall improvement, but significantly improve precision for the first few documents retrieved. MDS001 was our most successful run.

	Precision at N documents					Avg. prec.	Avg. (R) prec.
	5	10	20	30	200		
C-standard:							
Documents	0.1513	0.1423	0.1135	0.1051	0.0308	0.1534	0.1247
Pages	0.1680	0.1654	0.1192	0.1026	0.0281	0.1766	0.1677
TextTiles	0.1603	0.1577	0.1192	0.0974	0.0267	0.1817	0.1794
C-pivot	0.1859	0.1731	0.1212	0.0897	0.0287	0.1755	0.1459
150 words	0.1936	0.1731	0.1346	0.1115	0.0281	0.2416	0.2133
350 words	0.1923	0.1731	0.1288	0.0987	0.0275	0.2219	0.2130
Passages:							
C-standard	0.1949	0.1846	0.1385	0.1039	0.0288	0.2002	0.1640
C-pivot	0.1872	0.1885	0.1365	0.1077	0.0277	0.2268	0.2151
C-scaled	0.1821	0.1654	0.1327	0.1103	0.0308	0.1984	0.1463

Table 2: *Experiments with FR, disks 2 and 4, queries 251–300. C-pivot was the FR component of the run submitted as MDS001.*

	Precision at N documents					Avg. prec.	Avg. (R) prec.
	5	10	20	30	200		
C-standard:							
Documents	0.3480	0.3480	0.3090	0.2680	0.1288	0.1798	0.2199
Passages:							
C-pivot	0.3707	0.3608	0.3050	0.2760	0.1221	0.1804	0.2292

Table 3: *TREC 5 experiments, disks 2 and 4, queries 251–300. C-pivot was the run submitted as MDS001.*

3 Combination of Evidence

Term expansion is a popular and successful method to improve retrieval performance. There are many strategies employed, however many can be related to Rocchio’s formula for relevance feedback[4]:

$$\alpha(\text{Original Query Vector}) + \beta(\text{Average of Relevant Document Vectors}) - \gamma(\text{Average of Irrelevant Document Vectors})$$

Each of α , β , and γ are non-negative real numbers. Typical values might be $\alpha = 2$, $\beta = 1$ and $\gamma = 0$. Rather than using relevance assessments, it is often assumed that the top N documents are relevant, and these documents only are used in term expansion. Instead of regarding this information as a set of vectors to average, we may regard terms from a query, and terms from documents as two different sorts of evidence – evidence that can be combined.

As with our experiments in TREC4, we started with retrieval runs based on a cosine measure and a measure derived from the Okapi experiments. For the cosine we used:

$$\text{COS}(Q, D) = \frac{\sum_{t \in q \wedge d} (w_{q,t} \cdot w_{d,t})}{\sqrt{(\sum_{t \in q} w_{q,t}^2 \cdot \sum_{t \in d} w_{d,t}^2)}}$$

with $w_{q,t} = \log(N/f_t) + 1$ and $w_{d,t} = \log(f_{d,t} + 1)$. For Okapi we used:

$$OKA(Q, D) = \left(\frac{N - f_t}{f_t} \right) \left(\frac{f_{d,t}}{f_{d,t} + \sqrt{f_D}/av(\sqrt{f_D})} \right)$$

where $f_{x,t}$ is the frequency of t in x , and f_D is the length of D . N is the number of documents in the collection, and f_t is the number of documents containing t .

Again we were interested in only very short queries. Thus we used the description field only. After stopping and stemming, there were on average 8 terms per query.

The top 15 documents for each query were obtained using the COS measure. The text of these documents was agglomerated and the 45 best terms were extracted according to the formula:

$$f_{15d,t}/\log(f_t + C)$$

where $f_{15d,t}$ is the frequency of t in the top 15 documents, and $C = 20$, a constant to ensure only statistically significant variation was taken into account. We call this new query, Q' . Using these new queries, without using the original terms unless they were in the top 45 terms, a ranking was obtained using the COS measure given above.

The results of the two cosine measures were combined. The combination was based on a sum of angles. The method used was to weight the sum so that the original measure was given four times as much weight as the expanded query. This is a rough reflection of the weights used by others when applying, say, the Rocchio measure. If we let $COS1 = COS(Q, D)$, the cosine of the angle between the description field and the document, and let $COS2 = COS(Q', D)$, the cosine of the angle between the expansion terms and the document, the formula is:

$$Comb1(Q, D) = \frac{(COS1 + \alpha COS2)}{((COS1 + \alpha COS2)^2 + (((1 - COS1^2) + \alpha(1 - COS2^2))^2)^{0.5})^{0.5}}$$

The geometric interpretation is that this value is the cosine of the angle between the document vector and a vector obtained by adding the normalized original query vector, and α times the normalized derived query vector. We set $\alpha = 0.25$. We now had a result based on a short original query and a derived expanded query based on the database.

In TREC5 we had found good results for short queries by combining results obtained from cosine based measures and Okapi based measures. However, we no longer have two angles to combine, nor are values actually comparable, so we had first to normalize the Okapi results. Thus for each query, we multiplied the Okapi based similarity measure by the top score for the combined cosine measure and divided by top Okapi score. This guaranteed that the top score for both measures was the same. Thus $OKA2(Q, D) = \beta_Q OKA(Q, D)$.

Because more work had been put into developing a high quality result for the cosine, compared to the Okapi measure, we gave twice the weight to the cosine combined measure compared to the Okapi measure. Even though the Okapi measure was not a cosine, it could be treated as such so we used the same combination method as before. Thus we calculated:

$$\frac{(Comb1 + \gamma OKA2)}{((Comb1 + \gamma OKA2)^2 + (((1 - Comb1^2) + \gamma(1 - OKA2^2))^2)^{0.5})^{0.5}}$$

Here $\gamma = 0.5$. This run was submitted as MDS002.

In our manual runs, we simply added more evidence! We took the narrative field, and manually extracted what appeared to be low content phrases. If we had a sophisticated

	Precision at N documents					Avg.	Avg. (R)
	5	10	20	30	200	prec.	prec.
Desc	0.2440	0.2200	0.1850	0.1660	0.0878	0.1027	0.1397
Expand	0.2520	0.2140	0.1900	0.1807	0.0972	0.1168	0.1515
Comb1	0.2480	0.2380	0.2010	0.1907	0.1059	0.1229	0.1624
Desc(Oka)	0.1840	0.1640	0.1520	0.1393	0.0751	0.0755	0.1105
MDS002	0.2453	0.2320	0.2160	0.2013	0.1044	0.1214	0.1606
Narr	0.3480	0.2960	0.2850	0.2533	0.1193	0.1654	0.1938
Comb2	0.3800	0.3240	0.2750	0.2440	0.1184	0.1526	0.1957
Comb3	0.3240	0.2960	0.2730	0.2513	0.1233	0.1558	0.2016
MDS003	0.3060	0.3060	0.2780	0.2480	0.1187	0.1480	0.1875

Table 4: *TREC 5 combination experiments.*

phrase deletion algorithm, this would again be an automatic run. We then applied standard stopping and stemming to the narratives to have a remaining set of approximately 26 words per query, approximately 4 times larger that we had using the descriptions only. The strategy of combination was as follows:

1. Combine original description query with manually altered narrative query, $\alpha = 1.0$ (Row Comb2 in Table 4.)
2. Combine result with COS2, the result from modified queries obtained earlier, $\alpha = 0.25$ (Row Comb3 in Table 4.)
3. Combine result with OKA2, the normalized Okapi results, $\alpha = 0.5$

This result was submitted as MDS003.

For all runs, only the top 1,000 documents were used for processing, combining, or evaluating. There may be some benefit to using more documents in the base runs, but we expect that the effect would be extremely marginal. The results of these experiments are given in Table 4.

4 Chinese Retrieval

In the Chinese language each character represents a syllable. “Words” usually consist of one, two, or three syllables. Our past research [3] has concentrated on looking at whether individual characters or complete words are more effective as the basic indexing and retrieval units for IR. Complete words usually have a much more specific meaning than individual characters. For example when the character “dian” (electric) combines with “nao” (brain) the resultant word is “computer”. Consequently we had initially expected that word based retrieval would be more effective than character based.

Our experiments were run using *mg*, which has been used extensively for English language information retrieval. Because *mg* is not set up to handle 16 bit characters, we pre-processed the documents and queries to turn the text into ASCII strings. Each Chinese character was converted to a four character ASCII string, using a hexadecimal representation. Thus the sixteen bit code $a01c01c_{16}$ was converted to the character string “a01c”. This was the only

pre-processing done for character retrieval. Indexing and retrieval was then done using both the cosine and Okapi similarity measures. (Cosine-char and Okapi-char in Table 5.)

In Chinese there is no white space between words such as occurs in English. Consequently character sequences must first be *segmented* into words before word based indexing can be performed. In the past successful segmentation has been thought to be a significant impediment to the indexing and hence retrieval of Chinese text [7]. The segmentation problem in Chinese has been extensively researched, both in relation to information retrieval [7] and for other applications, such as speech recognition. No algorithm has been developed that has achieved perfect results. Indeed, much Chinese humour derives from the fact that some phrases may be legitimately segmented in different ways, giving rise to completely different meanings. The best segmentation schemes yield about 96% accuracy [6].

We used a dictionary based parsing method to segment the text. Dictionary methods may work by finding the longest substring in a sequence of text which is contained in a lookup dictionary. Such an approach is known as greedy parsing. We have used this method in previous work, with satisfactory results. Once again indexing and retrieval was then done using both the cosine and Okapi similarity measures. (Cosine-word and Okapi-word in Table 5.)

In the set of combination experiments, all runs were normalized so that the maximum score for one run for a particular query was made equal to the maximum score for the other run for that query. As well, a more common weighted sum was used:

$$\frac{W1 + \alpha W2}{(1 + \alpha)}$$

We combined as follows:

1. Combine character based cosine run with character based Okapi run, $\alpha = 0.5$

This result was submitted as MDS004.

1. Combine word based cosine run with word based Okapi run, $\alpha = 0.5$
2. Combine character based combined run with word based combined run, $\alpha = 0.5$

This result was submitted as MDS005.

The results are shown in Table 5. The performance is awful. After a lot of investigation we have still not found the bug in our system.

References

- [1] M.A. Hearst and C. Plaunt. Subtopic structuring for full-length document access. In R. Korfhage, E. Rasmussen, and P. Willett, editors, *Proceedings of the 16th Annual International Conference on Research and Development in Information Retrieval*, pages 59–68, Pittsburg, U.S.A., June 27 – July 1 1993. ACM.
- [2] A. Moffat, R. Sacks-Davis, R. Wilkinson, and J. Zobel. Retrieval of partial documents. Technical report, Key Centre for Knowledge Based Systems, Departments of Computer Science, RMIT and the University of Melbourne, Melbourne, Australia, 1993.

	Precision at N documents					Avg.	Avg. (R)
	5	10	20	30	200	prec.	prec.
Cosine-char	0.0842	0.1316	0.1158	0.1053	0.0539	0.0369	0.1021
Okapi-char	0.0105	0.0105	0.0184	0.0158	0.0116	0.0043	0.0213
Cosine-word	0.1368	0.1316	0.1237	0.1053	0.0661	0.0408	0.1066
Okapi-word	0.0842	0.1053	0.0921	0.0842	0.0497	0.0251	0.0798
Combined-char	0.0947	0.1158	0.0921	0.0930	0.0476	0.0268	0.0867
Combined-word	0.1368	0.1474	0.1211	0.1070	0.0655	0.0406	0.1068
Combined-all	0.1474	0.1211	0.1237	0.1123	0.0558	0.0371	0.1045

Table 5: *Chinese experiments. Combined-char was submitted as MDS004, Combined-all was submitted as MDS005.*

- [3] V. B. H. Nguyen, P. Vines, and R. Wilkinson. A comparison of morpheme and word based document retrieval for asian languages. In *International Conference on Database and Expert System Applications — DEXA 96*, page To Appear, Zurich, Switzerland, 1996.
- [4] J. J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART RETRIEVAL SYSTEM*, pages 243–264. Prentice Hall, New Jersey, 1966.
- [5] A. Singhal, C. Buckley, and M. Mitra. Pivoted document length normalization. In H.-P. Frei, D. Harman, P. Schauble, and R. Wilkinson, editors, *Proceedings of the 19th Annual International Conference on Research and Development in Information Retrieval*, pages 21–29, Zurich, Switzerland, August 18–22 1996. ACM.
- [6] R. Sproat and C.L. Shih. A statistical method for finding word boundaries in Chinese text. *Computer Processing of Chinese and Oriental Languages*, 4(4):336–351, 1990.
- [7] Z. Wu and G. Tseng. Chinese text segmentation for text retrieval: Achievements and problems. *Journal of the American Society for Information Science*, pages 532–542, October 1993.

SPIDER Retrieval System at TREC-5

Jean-Paul Ballerini, Marco Büchel, Ruxandra Domenig,
Daniel Knaus, Bojidar Mateev, Elke Mittendorf,
Peter Schäuble, Páraic Sheridan, Martin Wechsler

Swiss Federal Institute of Technology (ETH)
CH-8092 Zürich, Switzerland

Abstract

The ETH group participated in this year's TREC in the following tracks: automatic adhoc (long and short), the manual adhoc, routing, and confusion. We also did some experiments on the chinese data which were not submitted.

While for adhoc we relied mainly on methods which were well evaluated in previous TRECs, we successfully tried completely new techniques for the routing task and the confusion task: for routing we found an optimal feature selection method and included co-occurrence data into the retrieval function; for confusion we applied a robust probabilistic technique for estimating feature frequencies.

1 Introduction

ETH's contribution to TREC-5 consists of a new routing approach and robust probabilistic technique for the confusion track. Both approaches have been tuned by means of an essentially known reference method (Lnu.ltn, phrase indexing, query expansion). For comparison reasons, this reference method has been applied to the adhoc task and the results were submitted as official runs.

For the routing task, we focused on feature selection and using co-occurrence data. The latter approach evolved from an approach which has been successfully used in image synthesis. The basic idea is that a synthetic surface and a real surface look similar if their second order statistics are similar. In computer graphics this relationship is used to generate artificial surfaces, e.g. for virtual surgery [Hug, 1996].

Our contribution to the confusion track mainly relied on earlier projects where we learned how to deal with corrupted data. In speech retrieval phoneme recognition rates of 50% are observed when speech recognition is independent of the speakers [Schäuble et al., 1997]. Optical Character Recognition (OCR) may achieve word recognition rates of about 70% in the case of old and partially handwritten documents [Mittendorf et al., 1995]. Fortunately, low recognition rates do not necessarily imply poor retrieval effectiveness. It was shown that the effects of recognition error may cancel out to certain extent if the documents are long enough [Mittendorf & Schäuble, 1996].

The paper is structured as follows. We first describe the reference method applied to adhoc test (Section 2). Our approaches to the routing and confusion problem are described in Section 3 and 4. Our first attempt to Chinese text retrieval is described in Section 5.

In our terminology we denote the TREC disks, disk2, disk3, disk4, by D2, D3, and D4, last years routing data is denoted by R4 (mainly consisting of net data), and this years routing data is denoted by R5 (consisting of FBIS). For the definitions of our symbols refer to the symbols table in appendix A.

2 Adhoc Retrieval

For comparison reasons we applied our *reference method* to the adhoc task and submitted the results as official runs. For the adhoc task our group mainly adapted the methods reported by the SMART group in last year's TREC [Buckley et al., 1996], namely the Lnu.ltn weighting scheme, "automatic relevance feedback" by top ranked documents

for automatic adhoc and manual feedback for manual adhoc. Both feedback methods use Rocchio reweighting. We also implemented phrase indexing. We modified these well-known methods slightly by changing the feature selection method for the Rocchio reweighting and including an indexing capability to index both, British and American English simultaneously.

Indexing

Our indexing vocabulary $\Phi := \Phi_1 \cup \Phi_2$ consists of single words reduced by the Porter algorithm [Porter, 1980] $\varphi_h \in \Phi_1$ called the one-word features, and 2-word-phrases $\varphi_i \in \Phi_2$, which will be called 2-word features. A valid single word has a minimal length of three characters. The removal of important words shorter than this is prevented by using an anti-stopword list [Knaus et al., 1995]. Also, stopwords are removed [Salton, 1971]. A potential 2-word-phrase consists of any pair of adjacent non-stopwords as proposed in [Buckley et al., 1992]. The final list of phrases was composed using only phrases with a document frequency greater than 25 in Disk2 and Disk3 (D2D3) of the TREC collection. Table 1 shows the number of phrases for different minimal document frequencies. As explained later in this section, phrases do not contribute to the document length in the weighting schemes we used.

minimal df	# 2-word features
1	7'000'000
2	2'700'000
5	1'000'000
25	200'000
50	99'000

Table 1: Number of 2-word features for various document frequencies

For documents written in British English we introduced an additional procedure that maps single words into American English. For British-American synonyms (e.g. film-movie, queue-line) we constructed a lookup table (41 words) that was applied before reducing words. Another lookup table (272 words) was used after word reduction to account for different spellings (e.g. analyse-analyze, colour-color). Both lists were collected from various sources in the World Wide Web.

Basic Weighting Scheme

The Retrieval Status Values (RSV) are obtained by

$$\text{RSV}(q, d_j) = \sum_{\varphi_i \in \Phi(q, d_j)} a_{ij} \cdot b_i, \quad (1)$$

where $\Phi(q, d_j)$ denotes the set of 1-word or 2-word indexing features occurring in both the query q and the document d_j , a_{ij} denotes the weight of the document feature and b_i denotes the weight of the query feature.

For the (automatic and manual) adhoc tracks as well as for baseline methods for routing and confusion tracks we used *Lnu.ltn* weighting [Singhal et al., 1996] as our basic weighting scheme. Subsequently $\text{ff}(\varphi_i, d_j)$ denotes the feature frequency (number of occurrences) of φ_i in d_j and $\text{nidf}(\varphi_i)$ denotes the normalized inverse document frequency $\text{nidf}(\varphi_i) := 1 - \frac{\log(\text{df}(\varphi_i)+1)}{\log(n+1)}$, where n denotes the number of documents in the collection and $\text{df}(\varphi_i)$ denotes the document frequency (the number of documents φ_i occurs in). The document length $l^{\Phi_1}(d_j)$ is the number of one-word features only, whereas the average feature frequency ($\overline{\text{ff}}^{\Phi}(d_j)$) is always the average over all features (regardless of the feature type):

$$a_{ij} = \frac{1}{(1-s) \cdot p + s \cdot l^{\Phi_1}(d_j)} \cdot \frac{1 + \log(\text{ff}(\varphi_i, d_j))}{1 + \log(\overline{\text{ff}}^{\Phi}(d_j))} \quad (2)$$

$$b_i = \text{nidf}(\varphi_i) \frac{1 + \log(\text{ff}(\varphi_i, q))}{1 + \log(\overline{\text{ff}}^{\Phi}(q))}, \quad (3)$$

where s denotes the slope and p denotes the pivot. The pivot is computed as the average document length ($p = \overline{l}^{\Phi_1}(d_j)$).

Query Expansion with Relevance Feedback and Feature Selection

Depending on the task there are different *sources of relevance information*. We denote the set of documents assumed to be relevant as D^{rel} . For both automatic adhoc and manual adhoc the same relevance feedback method is applied given D^{rel} .

- **Automatic Adhoc**

In a first pass all original queries are evaluated without any feedback information and ranked lists are retrieved. We then assume that the r top ranked documents are relevant. In the second pass the original queries are expanded by using this relevance information and the 1000 top ranked documents are determined.

- **Manual Adhoc**

An interactive manual process done by the users determines a set of relevant documents for each topic. The queries are formulated by the user and may be modified during the searching process.

The system used allowed relevance feedback searching as well as highlighting 1-word and 2-word features. The users were requested to justify their relevance judgements for each document. Each topic was investigated by only one user, therefore the relevance judgements are strongly user dependent.

The user population was highly homogeneous since all of them are non-native English speakers and work in the field of information retrieval.

No time constraints were set.

Given the set of relevant documents we *select h features* from the original query q and the relevant documents. Let $\Phi^{sel} = \Phi(q) \cup \bigcup_{d_j \in D^{rel}} \Phi(d_j)$ be the set of features from the original query and from all relevant documents. All features of Φ^{sel} are ranked by the following method (called *ltn-selection*) and the h top ranked features are selected to build together with all original query features the new query q' .

$$sel_{ltn}(\varphi_i) = \alpha \cdot b_i + \beta \cdot \frac{1}{|D^{rel}|} \cdot \sum_{d_k^{rel} \in D^{rel}} b_{ik}^{rel} \quad (4)$$

where for all $d_k^{rel} \in D^{rel}$

$$b_{ik}^{rel} = \begin{cases} \text{idf}(\varphi_i) \frac{1 + \log(\text{ff}(\varphi_i, d_k^{rel}))}{1 + \log(\text{ff}(\Phi^{sel}(d_k^{rel})))} & : \varphi_i \in d_k^{rel} \\ 0 & : \text{else} \end{cases}$$

The normalisation by the average feature frequency $\log(\text{ff}(\Phi^{sel}(q)))$ of the original query is needed to have the weights of original query features and the weights of feedback document features on the same scale.

We also tried other feature selection methods, but they all performed worse (in terms of effectiveness). The only one that performed nearly equally well was the df_{rel} -selection where the number of documents in D^{rel} containing the feature was used to rank the features ($sel_{df}(\varphi_i) = |\{d_k^{rel} \in D^{rel} | \varphi_i \in d_k^{rel}\}|$). This is the feature selection method reported in [Buckley et al., 1996].

After query expansion the following Rocchio reweighting was applied to the expanded query q' :

$$\text{RSV}^{\text{feedback}}(q', d_j) = \sum_{\varphi_i \in \Phi(q', d_j)} b'_i \cdot a_{ij} \quad (5)$$

where

$$b'_i = sel_{ltn}(\varphi_i)$$

Final runs

The TREC-4 data has been used to optimize some of the parameters. For TREC-5 we submitted the following experiments:

	\geq Median	$<$ Median	Average Precision (over all topics)
automatic short	25	25	0.1726
automatic long	41	9	0.2425
manual feedback	43	7	0.3165

Table 2: ETH Results for Adhoc Tasks.

- **ETHal1**

automatic adhoc: queries 251-300 versus D2D4, long queries (all fields),
automatic query expansion with $r = 20$,
feature selection with *ltn*-selection, $h = 50, \alpha = 1.0, \beta = 1.0$
Lnu.ltn-weighting with slope $s = 0.24$, top 1000 ranks retrieved.

- **ETHas1**

automatic adhoc: queries 251-300 versus D2D4, short queries (description field only),
automatic query expansion with $r = 20$,
feature selection with *ltn*-selection, $h = 50, \alpha = 1.0, \beta = 1.0$
Lnu.ltn-weighting with slope $s = 0.24$, top 1000 ranks retrieved.

- **ETHme1**

manual adhoc: queries 251-300 versus D2D4, long queries (all fields),
manual search for relevant documents,
query expansion with all found relevant documents,
feature selection with *ltn*-selection, $h = 50, \alpha = 1.0, \beta = 1.0$
Lnu.ltn-weighting with slope $s = 0.24$, top 1000 ranks retrieved.

Results We report the results of the final runs in table 2. As expected, the automatic long method had better results than the automatic short method, even though some topic descriptions could have been misleading.

The manual feedback improved the average precision significantly. This indicates, that—in contrast to last year’s interactive TREC—the user’s and the assessor’s relevance judgements were rather similar although our users were non-native English speakers. In addition, the lack of any time constraints enables the searcher to search in his/her own pace without any hurry. However, this is comparable to a real user’s search-task, where the search is finished, when the user does not expect the system to retrieve any more relevant information. For these experiments users needed 30 to 40 minutes per topic.

3 Routing

For the first time the ETHZ group attacked the routing problem seriously. We investigated two problems: optimal feature selection and the use of co-occurrence data in the routing environment.

3.1 Feature Selection

We evaluated feature selection methods that are based on contingency tables:

	$\#\{d_j d_j \in R\}$	$\#\{d_j d_j \notin R\}$	
$\#\{d_j \varphi_i \in d_j\}$	x_{11}	x_{12}	$x_{1.}$
$\#\{d_j \varphi_i \notin d_j\}$	x_{21}	x_{22}	$x_{2.}$
	$x_{.1}$	$x_{.2}$	$x_{..}$

We used the U-measure that is defined for the U-test [Andersen, 1992] for feature selection:

$$\mu_U(\varphi_i) := \sqrt{x_{..} \frac{x_{11}x_{22} - x_{12}x_{21}}{x_{1.}x_{.1}x_{2.}x_{.2}}}. \quad (6)$$

Note the similarity to the χ^2 -measure,

$$\mu_{\chi^2}(\varphi_i) := x_{..} \frac{(x_{11}x_{22} - x_{12}x_{21})^2}{x_{1.}x_{.1}x_{2.}x_{.2}}. \quad (7)$$

That means, that

$$\mu_U(\varphi_i) = \begin{cases} \sqrt{\mu_{\chi^2}(\varphi_i)} & x_{11}x_{22} \geq x_{12}x_{21} \\ -\sqrt{\mu_{\chi^2}(\varphi_i)} & x_{11}x_{22} < x_{12}x_{21}. \end{cases} \quad (8)$$

Experiments which used the χ^2 -measure for feature selection showed only moderate results [Schütze et al., 1995]. We briefly explain why the U-measure is well suited for feature selection, and certainly better suited than the χ^2 -measure. The χ^2 -measure rewards positive *and* negative correlation, i.e. μ_{χ^2} yields large values if $x_{11}x_{22} \ll x_{12}x_{21}$ *and* if $x_{11}x_{22} \gg x_{12}x_{21}$. The measure μ_U only rewards positive correlation. Thus a feature whose absence implies relevance or for which presence implies irrelevance is a good χ^2 -feature but a bad U-feature (negative numbers with large absolute values). Since almost all retrieval functions do not make use of negative evidence the U-measure is the better feature selection method for most retrieval methods.

Experiments with U-features

We evaluated $\mu = \mu_U$, $\mu = \mu_{\chi^2}(\varphi_i)$ and several other measures μ (Robertson Selection Value [Robertson et al., 1995], F4-formula [Eftimiadis, 1993], measure used in [Allan et al., 1996], etc.) on the contingency table with the following experimental setting:

1. Build contingency tables $(x_{11}, x_{12}, x_{21}, x_{22})$ on the training data for all features which occur in at least one relevant document ($x_{11} \geq 1$), do not occur in at least one irrelevant document ($x_{22} \geq 1$), occur in at least 15 documents ($x_{1.} \geq 15$), and do not occur in at least 15 documents ($x_{2.} \geq 15$),
2. Rank all features according to $\mu(\varphi_i)$.
3. Select the N_1 top ranked 1-word features $\varphi_l \in \Phi_1$ and the N_2 top ranked 2-word features $\varphi_k \in \Phi_2$ and let $q := \{\varphi_{l_0}, \dots, \varphi_{l_{N_1-1}}, \varphi_{k_0}, \dots, \varphi_{k_{N_2-1}}\}$.
4. Rank the documents of the test collection according to $RSV_{Lnu.ltn}(q, d_j)$.

We used the TREC-4-routing queries and D2D3 for training and the TREC-4-routing data (R4) for testing. The features selected by μ_U consistently produced the best average precision values or values that were among the best for different query sizes (N_1, N_2) , the experiments are reported in [Mateev, 1996].

We chose $N_1 := 50$ and $N_2 := 20$ which yields an average precision of 0.3103 with $RSV_{Lnu.ltn}$ on R4, we will denote this method as $RSV_{Lnu.ltn:50:20}$. We found out later that $N_1 := 60$ and $N_2 := 30$ yield slightly better results with the same experimental setting (0.3172).

The U-measure is a measure that is normalized over all marginal values (i.e. $(x_{1.}, x_{.1}, x_{2.}, x_{.2})$). Therefore it is predestined to rather select all features above a certain threshold rather than a fixed number of top ranked features. However, we did not experiment with such thresholds.

3.2 Using Co-occurrence Data for Routing Retrieval

There are several attempts reported in information retrieval literature that make use of the co-occurrence of features, see e.g. [van Rijsbergen, 1977]. These attempts brought only slight improvements if any at all. However, none of the previous approaches were able to use as much training data as one is able to use for the routing task. We therefore felt encouraged to attempt to use the co-occurrence of features for the routing task.

Description of Co-occurrence Indexing and Retrieval

We tried a straightforward approach of generalizing the Binary Independence Retrieval model (BIR) and the Robertson—Spärck Jones weighting formula.

1. Choose a sequence of feature distances e.g. $\Delta^{(0)} := 0$, $\Delta^{(1)} := 1$, $\Delta^{(2)} := 10$, $\Delta^{(3)} := 30$, $\Delta^{(4)} := 200$.

2. Fix a set of N 1-word features $\varphi_i \in \Phi_1$.

3. Define the description matrix (in generalization of a description vector) $d_j^{(i)} := (a_{j,lk}^{(i)})_{l,k=0}^{N-1}$, such that

$$a_{j,lk}^{(i)} := \begin{cases} 1 & \text{if there exists an occurrence of } \varphi_k \text{ at least } \Delta^{(i-1)} + 1 \\ & \text{and at most } \Delta^{(i)} \text{ tokens after an occurrence of } \varphi_l, \\ 0 & \text{else.} \end{cases} \quad (9)$$

Note that the matrices $d_j^{(i)}$, $i = 1, \dots, 4$ are usually not symmetric, and that the matrix $d_j^{(0)}$, is a diagonal matrix with the diagonal describing the binary description vector of a document used for probabilistic retrieval (in this case we define $\Delta^{(-1)} := -1$).

4. Define the query weight matrices $q^{(i)} := (b_{lk}^{(i)})_{l,k=0}^{N-1}$,

$$b_{lk}^{(i)} := \log \frac{p_{lk}^{(i)}(1 - q_{lk}^{(i)})}{q_{lk}^{(i)}(1 - p_{lk}^{(i)})}, \quad (10)$$

where $p_{lk}^{(i)} := P(d_{j,lk}^{(i)} = 1 | R)$ and $q_{lk}^{(i)} := P(d_{j,lk}^{(i)} = 1 | \bar{R})$. The probabilities $p_{lk}^{(i)}$ and $q_{lk}^{(i)}$ are estimated by relative frequencies on the training data. To get robust estimates even for zero-values, we changed to the so-called 0.5-formula, see e.g. [Robertson et al., 1995].

5. Define Retrieval Status Values:

$$\text{RSV}^{(i)}(q, d_j) := \text{Tr}(d_j^{(i)} q^{(i)T}), \quad (11)$$

where Tr denotes the trace of a matrix and A^T denotes the transposition of matrix A .

To clarify the ideas behind this co-occurrence retrieval methods we make the following remarks.

- The $\text{RSV}^{(0)}$ is exactly the retrieval status value of the BIR, $\text{RSV}^{(1)}$ comprises a retrieval based on a kind of “phrase indexing”.
- The choice of $\Delta^{(0)}, \dots, \Delta^{(4)}$, is very intuitive. The vague idea was to index the original terms $\Delta^{(0)}$, index phrases $\Delta^{(1)}$, index the co-occurrence in a local region of sentence length $\Delta^{(2)}$, of paragraph length $\Delta^{(3)}$, and of average document length $\Delta^{(4)}$.
- None of the retrieval status values makes use of feature frequency information, but implicitly the information of feature frequency is used in $\text{RSV}^{(1)}$, $\text{RSV}^{(2)}$, $\text{RSV}^{(3)}$, and $\text{RSV}^{(4)}$ since a feature with high frequency has a better chance to co-occur with another feature.
- Since we do not explicitly use feature frequency information, document normalization is not necessary (though a good normalization might help).
- Each RSV can be perfectly motivated in the framework of probabilistic retrieval.
- Since only co-occurrences in a window of predefined length are used, the co-occurrence indexing carries valuable local information.

Combination

Since each $\text{RSV}^{(i)}$ is based on a different kind of indexing, we can expect the descriptions to be rather independent. Thus $\text{RSV}^{(0)}, \dots, \text{RSV}^{(4)}$ proffer themselves for combination. Since in our preliminary experiments $\text{RSV}^{(4)}$ did not produce good results we only combined $\text{RSV}^{(0)}, \dots, \text{RSV}^{(3)}$, and $\text{RSV}_{Lnu.ltn:50:20}$, but $\text{RSV}^{(4)}$ was still used for training of the combination parameters. We combined the RSV-values linearly and determined the combination parameters with logistic regression on a per query basis, where the combination values were trained on D1D2D3. The result of this combination was submitted as our final run **ETHru1**.

Data	$RSV_{Lnu.ltn:50:20}$	N	$RSV^{(0)}$	$RSV^{(1)}$	$RSV^{(2)}$	$RSV^{(3)}$	$RSV^{(4)}$
R4	0.3103	20	0.2867	0.2803	0.3099	0.3073	0.2835
R4	—	60	0.2646	0.2976	0.3168	0.3090	0.2610
R5	0.2046	20	0.1839	0.1578	0.1915	0.1863	0.1812

Table 3: Average Precision of 45 Topics vs R5 (FBIS) and of 50 Topics vs R4 (TREC-4 Routing Data)

Data	Average Precision	Percent improvement over Lnu.ltn:50:20
R4	0.3405	10%
R5	0.2401	17%

Table 4: Combination Results

Experiments

We trained the matrices $q^{(i)}$ on D1D2D3. Since at the moment the SPIDER retrieval system does not administer position information we did not have the chance to compute $RSV^{(i)}$, $i = 1, \dots, 4$ on the complete set of documents. So we preselected a set of 1000 Documents for each query by computing the $RSV_{Lnu.ltn:50:20}$. We re-ranked the documents according to $RSV^{(0)}, \dots, RSV^{(4)}$. The average precision values for each of the RSV is reported in table 3.

The results of the combination are reported in table 4.

Table 5 shows how the co-occurrence ranking method does compares with other submitted methods. We report the number of topics that are above or below the median. In brackets we report the numbers for the 39 queries with more than two relevant documents.

Analyzing Co-occurrence Indexing and Retrieval

- The described retrieval method is a good retrieval method for the routing environment though not one of the best.
- The average precision values in table 3 indicate that co-occurrence of terms is a very important source of information for retrieval. It is surprising that probabilistic retrieval on feature pairs that occur within 10 tokens ($RSV^{(2)}$) performs so significantly better than retrieval on conventional 1-word features ($RSV^{(0)}$).
- The occurrence of features ($RSV^{(0)}$), the co-occurrence within a region of more than one and at most 10 tokens ($RSV^{(2)}$), and the co-occurrence within a region of more than 10 and at most 30 tokens ($RSV^{(3)}$) are good indicators of relevance, whereas phrases and the co-occurrence of features that are more than 30 tokens apart from each other seem to be less well suited.
- The methods are well suited for combination (see table 4).
- Table 5 indicates that the preselection of the first 1000 documents is not good enough. A run of the co-occurrence ranking without a preselection would probably be better. However the preselection was only made because of the lack of position information in our index structure. A different implementation would make the preselection obsolete.

	Relevant Retrieved @ 1000	R-Precision	Average Precision
\geq Median	31 (25)	36 (30)	32 (30)
$<$ Median	14 (14)	9 (9)	13 (9)

Table 5: Comparison of Co-occurrence Ranking ETHru1 on 45 Topics (39 Topics)

We gained the conviction that the information about co-occurrence is a very valuable indicator of relevance. However, while developing our co-occurrence retrieval method we made a lot of ad-hoc decisions (selection of $\Delta^{(i)}$, document description with matrices, Robertson—Sparck Jones weighting, combination, etc.) that should be given further thoughts.

4 Confusion track

For the confusion track our goal was mainly to apply the probabilistic matching of terms that was developed for the retrieval of library cards [Mittendorf et al., 1995] and speech retrieval [Wechsler & Schäuble, 1995] to the different problem of retrieval of rather long OCR-corrupted documents in a monolingual collection. In addition to the probabilistic matching we hoped to be able to detect systematic errors for which we could compensate.

Since our probabilistic matching method detects a large number of similar query word occurrences, it is computationally too expensive to run the algorithms on all documents. Therefore, for each query, we preselected 2000 documents of the FR94-degrade5 (d5) collection and 2000 documents of the FR94-degrade20 (d20) collection by indexing documents with overlapping N-gram features (N consecutive alphanumeric characters with at most one separation character) and retrieving the document with the *Lnu.ltn*-weighting scheme. We decided to set $N := 4$ for the d5 collection and after having a short glimpse at d20 we thought that this collection might be too corrupted for 4-grams and set $N := 3$ for this collection. (After submitting our final runs we did some comparisons of the ranking on the perfect documents and the corrupted documents and we now think that the decision for 3-grams was probably too anxious.) We submitted these rankings as ETHD5N and ETHD20N.

Starting from those N-gram based ranking lists, we performed a re-ranking using the following strategy. We decided to take a weighting formula with pivoted byte size normalization, suggested by Singhal, Buckley, and Mitra [Singhal et al., 1996]

$$\text{RSV}(q, d_j) := \frac{1}{(1-s) \cdot p + s \cdot l^{\text{byte}}(d_j)} \sum_{\varphi_i \in q, \text{Eff}(\varphi_i, d_j) > 0} (1 + \log(\text{ff}(\varphi_i, q))) \cdot \text{idf}(\varphi_i) \cdot (1 + \log(\text{Eff}(\varphi_i, d_j))) \quad (12)$$

where $l^{\text{byte}}(d_j)$ denotes the length of the document in bytes. The corrupted $\text{idf}(\varphi_i)$ in d5 and d20 were substituted by the idf in last year's ad-hoc data D2D3. The slope was set to $s := 0.3$ and the pivot was set to the average byte size of documents in d5 or d20, $p := l^{\text{byte}}$. The values $\text{Eff}(\varphi_i, d_j)$ are supposed to be robust estimates of the feature frequency in the document, we will describe below how we get these estimates. For each query we performed the following steps:

- Index the query using stopword removal and Porter reduction [Porter, 1980]. For these experiments we did not apply phrase indexing for technical reasons.
- To each position in the document an accumulator is allocated that counts the number of matching characters between the substring starting at this position and the feature. To each character of the feature, the set of accumulators to be incremented is determined using a character index of the document and the character's relative position in the feature. This yields a set of potential feature beginnings. To account for insertion and deletion errors, we investigate the accumulators in a small range around the potential feature beginnings and increment the accumulator with the highest count. The accumulators values are then used to determine the beginnings of valid slots. A valid slot s_i must contain at least P percent of the feature's characters (P is referred to in Table 6). The slot end is either determined by aligning the last two characters or using the feature's length. Note that the features are reduced words with a starting delimiter, e.g. for the word *replacement* the string *_replac* is matched to the documents. This delimiter prohibits subword matching, which we think is useful for English (e.g. the word 'ring' should not match bothering).
- For each slot s_i , compute an edit distance $d(\varphi_i, s_i)$ between the slot and the feature. We used a distance function that has already been described in [Mittendorf et al., 1995].
- For each slot s_i , based on the edit distance, compute the probability p_i that s_i contains an occurrence of the feature using the edit distance. A reliable probability estimation function would require training data (perfect and corrupted data), which we did not have time to produce for the confusion track. So we applied a very crude

	P	d5		d20	
		α	res	α	res
nff	–	1.014	3.44	1.939	5.352
eff	50%	0.765	2.714	0.954	4.428
	60%	0.816	2.465	1.256	3.978
	70%	0.841	2.426	1.315	3.98
	75%	0.847	2.792	1.3602	4.191

Table 6: Residual Standard Error (res) of fitted lines for various P and α .

probability estimation:

$$p_l := \begin{cases} 1 & \text{if } \frac{d(s_l, \varphi_i)}{\text{len}(\varphi_i)} \leq 0.2 \\ 0 & \text{else,} \end{cases} \quad (13)$$

where $\text{len}(\varphi)$ is the number of characters in the query feature. The constant 0.2 is dependent on the edit distance function and was determined empirically. We will conduct more detailed analysis of the edit distance function and probability estimation.

- For each document and each feature, compute the expected feature frequency:

$$\text{eff}(\varphi_i, d_j) = \sum_{l=0}^{p-1} p_l. \quad (14)$$

- Search for systematic errors of the expected feature frequency eff and compensate for them:

$$\text{eff}(\varphi_i, d_j) \mapsto \text{Eff}(\varphi_i, d_j).$$

We selected 100 documents arbitrarily out of $d20$ (since $d20 \subset d5 \subset \text{FR94}$) and compared in the 100 triples of documents (FR94 , $d5$, $d20$) the feature frequencies ff (in FR94) with the noisy feature frequencies nff (feature frequencies in $d5$ and $d20$) and with the expected feature frequencies eff ($d5$ and $d20$). We fitted scatterplots by a line: $\text{ff}(\varphi_i, d_j) = \alpha \text{nff}(\varphi_i, d_j)$ and $\text{ff}(\varphi_i, d_j) = \alpha \text{eff}(\varphi_i, d_j)$, respectively. The results for different values for P (slot selection threshold) and α are reported in Table 6. To minimize the residual standard error (res), we decided to use the function

$$\text{Eff}(\varphi_i, d_j) := \alpha \text{eff}(\varphi_i, d_j),$$

with the following parameters:

$$d5 : P := 70\% \alpha := 0.8413 \quad (15)$$

$$d20 : P := 60\% \alpha := 1.2562. \quad (16)$$

Final runs

The final runs were submitted using the method described above as **ETHD5P** and **ETHD20P**. The pre-selection run on N-grams (4-gram and 3-gram) with `Lnu.ltn` weighting were submitted as **ETHD5N** and **ETHD20N**. A normal `Lnu.ltn` run with the basic method described in section 2 was submitted as **ETHFR94**.

Issues for improvement

The key part of this method is the probability estimation function, which was defined very crudely for this track. Alternative functions, maybe with additional information (e.g. the number of equal characters), could help to improve probability estimation. The edit distance function itself might be optimized using a character confusion matrix (which however is difficult to obtain).

On the other hand, the compensation $\text{eff}(\varphi_i, d_j) \mapsto \text{Eff}(\varphi_i, d_j)$ could be made dependent on recognition characteristics of the features, e.g. with the help of a confusion matrix on characters. Further experiments and analysis have to be done in this matter.

Results

We realized that we made a mistake applying the weighting formula (12) as it is since $\text{Eff}(\varphi_i, d_j) \in [0, 1]$ can happen. Because of the small compensation factor $\alpha = 0.8413$ for d5 and because of the logarithmic weighting negative weights can occur quite often. This is actually wrong and should not happen. This explains the bad results for ETHD5P (expected run length: 205.06, worse than the median over all submitted rankings). We did not yet do experiments with a better weighting scheme. However, we are sure that our approach is a good approach since for d20 the probabilistic method ETHD20P has the best result of all submitted rankings (expected run length: 115.41), in spite of the “wrong” weighting formula. For d20 we used a compensation factor $\alpha = 1.2562$ and thus there appeared no undesired negative weights.

We were surprised that ETHD5N, which was only thought to be a ranking for preselection, is the best submitted method for d5 (expected run length: 25.10). We will analyze the results further after the conference. In particular, we will adjust the weighting scheme.

5 Chinese Track

Another first for the ETHZ group this year involved running experiments on the Chinese data of the multilingual track, though these experiments were conducted after the submission deadline and so are unofficial. We used simple indexing techniques involving n-grams and then investigated various combinations of retrieval techniques available through the SPIDER system. These include the use of automatic relevance feedback using the top r ranked documents of an initial retrieval run and the use of similarity thesauri for query expansion.

To date we have indexed the Chinese collection using both 1-grams and 2-grams. Even with these two approaches we have noted a very significant difference in the resulting index. Indexing the 164,788 documents with 1-grams (each individual Chinese character becomes an indexing feature) gives an indexing vocabulary of 10,701 features. Using 2-grams results in a vocabulary of 2,089,778 features! This is without stopword removal since we do not yet have a list of Chinese stopwords.

On each of the two indexes, we have run a baseline experiment using cosine weighting (*lnc.ltn*) and experiments with automatic feedback with r set to each of 3, 5, and 10 (*ltn*-selection, $h = 50$, $\alpha = 1.0$, $\beta = 1.0$). Against these baselines we also performed experiments using the Lnu.ltn pivoted normalisation weighting scheme, both with and without automatic relevance feedback loops, plus a series of experiments using further query expansion through the use of similarity thesauri. Our most successful experimental setup on the TREC5 data was the use of Lnu.ltn weighting with an automatic feedback loop of the top ten retrieved documents expanded to 50 terms, and without further expansion through a similarity thesaurus. We hope to validate these results and explore further possibilities through participation in the TREC-6 conference.

Acknowledgements: The authors would like to thank Markus Flückiger and Daniel To for their help with the implementation, and the realization of the experiments.

A Symbols

Symbol	Variable Name	Definition
Φ_1		set of one word features
Φ_2		set of two word features (phrases)
Φ		$\Phi_1 \cup \Phi_2$
$l^\Phi(d_j)$	doc_len_all_feat_dj	length of document d_j counted as number of features, $l^\Phi(d_j) := \sum_{\varphi_i \in \Phi, \varphi_i \in d_j} 1$
$l^{\Phi_1}(d_j)$	doc_len_1w_feat_dj	length of document d_j counted as number of one word features, $l^{\Phi_1}(d_j) := \sum_{\varphi_i \in \Phi_1, \varphi_i \in d_j} 1$
$l^{\Phi_2}(d_j)$	doc_len_ph_feat_dj	length of document d_j counted as number of two word features, $l^{\Phi_2}(d_j) := \sum_{\varphi_i \in \Phi_2, \varphi_i \in d_j} 1$
$\overline{\text{ff}}^\Phi(d_j)$	avg_ff_all_feat_dj	average feature frequency in document d_j over all features, $\frac{1}{l^\Phi(d_j)} \sum_{\varphi_i \in \Phi} \text{ff}(\varphi_i, d_j)$
$\overline{\text{ff}}^{\Phi_1}(d_j)$	avg_ff_1w_feat_dj	average feature frequency in document d_j over all one word features, $\frac{1}{l^{\Phi_1}(d_j)} \sum_{\varphi_i \in \Phi_1} \text{ff}(\varphi_i, d_j)$
$\overline{\text{ff}}^{\Phi_2}(d_j)$	avg_ff_ph_feat_dj	average feature frequency over all two word features in document d_j , $\frac{1}{l^{\Phi_2}(d_j)} \sum_{\varphi_i \in \Phi_2} \text{ff}(\varphi_i, d_j)$
\overline{l}^Φ	avg_doc_len_all_feat	average of $l^\Phi(d_j)$
\overline{l}^{Φ_1}	avg_doc_len_1w_feat	average of $l^{\Phi_1}(d_j)$
\overline{l}^{Φ_2}	avg_doc_len_ph_feat	average of $l^{\Phi_2}(d_j)$
$l^T(d_j)$	doc_len_all_token_dj	length of document d_j counted as number of all tokens, $l^T(d_j) = \sum_{\varphi_i \in \Phi} \text{ff}(\varphi_i, d_j)$
$l^{T_1}(d_j)$	doc_len_1w_token_dj	length of document d_j counted as number of one word tokens, $l^{T_1}(d_j) = \sum_{\varphi_i \in \Phi_1} \text{ff}(\varphi_i, d_j)$
$l^{T_2}(d_j)$	doc_len_ph_token_dj	length of document d_j counted as number of two word tokens, $l^{T_2}(d_j) = \sum_{\varphi_i \in \Phi_2} \text{ff}(\varphi_i, d_j)$

References

- [Allan et al., 1996] Allan, J., Callan, J., Croft, W., & Lu, Z. (1996). Recent Experiments with INQUERY. In *TREC-4*.
- [Andersen, 1992] Andersen, E. (1992). *The Statistical Analysis of Categorical Data*. Springer, Berlin, third edition.
- [Buckley et al., 1992] Buckley, C., Salton, G., & Allan, J. (1992). Automatic Retrieval With Locality Information Using SMART. In *TREC-1 Proceedings*, pp. 59–72.
- [Buckley et al., 1996] Buckley, C., Singhal, A., & Mitra, M. (1996). New Retrieval Approaches Using SMART: TREC-4. In *TREC-4 Proceedings*.
- [Eftimiadis, 1993] Eftimiadis, E. (1993). A User-Centered Evaluation of Ranking Algorithms for Interactive Query Expansion. In *ACM SIGIR Conference on R&D in Information Retrieval*, pp. 146–159.
- [Hug, 1996] Hug, J. (1996). Analyse und Sybtgese der Textur von Organoberflächen. Master's thesis, Institute for Communication Systems.
- [Knaus et al., 1995] Knaus, D., Mittendorf, E., Schäuble, P., & Sheridan, P. (1995). Highlighting Relevant Passages for Users of the Interactive SPIDER Retrieval System. In *TREC-4 Proceedings*.
- [Mateev, 1996] Mateev, B. (1996). Stochastic Dependence of Indexing Features and the Routing Problem. Diploma Thesis, Department of Computer Science, ETH Zürich.
- [Mittendorf & Schäuble, 1996] Mittendorf, E., & Schäuble, P. (1996). Measuring the Effects of Data Corruption on Information Retrieval. In *Symposium on Document Analysis and Information Retrieval*, pp. 179–189.

- [Mittendorf et al., 1995] Mittendorf, E., Schäuble, P., & Sheridan, P. (1995). Applying Probabilistic Term Weighting to OCR Text in the Case of a Large Alphabetic Library Catalogue. In *ACM SIGIR Conference on R&D in Information Retrieval*, pp. 328–335.
- [Porter, 1980] Porter, M. F. (1980). An Algorithm for Suffix Stripping. *Program*, 14(3), 130–137.
- [Robertson et al., 1995] Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M. M., & Gatford, M. (1995). OKAPI at TREC-3. In *TREC-3 Proceedings*, pp. 109–126.
- [Salton, 1971] Salton, G. (1971). *The SMART Retrieval System-Experiments in Automatic Document Processing*. Prentice Hall, Englewood, Cliffs, New Jersey.
- [Schäuble et al., 1997] Schäuble, P., Sheridan, P., & Wechsler, M. (1997). Cross-Language Speech Retrieval. submitted for SIGIR'97.
- [Schütze et al., 1995] Schütze, H., Hull, D., & Pedersen, J. (1995). A Comparison of Classifiers and Document Representations for the Routing Problem. In *ACM SIGIR Conference on R&D in Information Retrieval*, pp. 229–237.
- [Singhal et al., 1996] Singhal, A., Buckley, C., & Mitra, M. (1996). Pivoted Document Length Normalization. In *ACM SIGIR Conference on R&D in Information Retrieval*, pp. 21–29.
- [van Rijsbergen, 1977] van Rijsbergen, C. (1977). A Theoretical Basis for the Use of Co-occurrence Data in Information Retrieval. *Journal of Documentation*, 33, 106–119.
- [Wechsler & Schäuble, 1995] Wechsler, M., & Schäuble, P. (1995). Speech retrieval based on automatic indexing. In Ruthven, I., editor, *Proceedings of the Final Workshop on Multimedia Information Retrieval (MIRO'95)*, Electronic Workshops in Computing, Glasgow. Springer.

Ad Hoc Experiments Using EUREKA

Allan Lu, Maen Ayoub, Jianhua Dong

Lexis-Nexis

A member of the Reed Elsevier plc group

9443 Springboro Pike

Miamisburg, OH 45342

(allan.lu, maen.ayoub)@lexis-nexis.com

1.0 Introduction

Our research for TREC5 focused on search and retrieval of full-text documents with short natural language (NL) queries. It has been our strong belief that the queries submitted to any operational retrieval system, especially those on the Internet, are short or very short, and that an effective approach to processing short NL queries has great application potential. We also looked at data fusion [1] with the assumption that a number of well-developed and specialized retrieval functions would probably outperform a single well-developed but general function. For example, two functions, one specialized in retrieving medium to long documents and another short to medium documents, would deliver better performance if they could be combined properly. Finally, we investigated the problem of selecting documents for relevance feedback. Unhappy with the assumption that all of the top 20 retrieved documents, for example, are relevant and ready for a relevance feedback process, we revisited the cluster hypothesis [2] and experimented with clustering the top 20 documents and automatically selecting a subset for relevance feedback.

Our research system named EUREKA (End User Research Enquiry and Knowledge Acquisition) was used for carrying out the experiments. EUREKA consists of a rich set of UNIX tools which can be assembled into various automatic indexing and ranking/filtering mechanisms, either as a new retrieval system or as a simulation of an interesting research system. The tool set design provides a maximum level of flexibility.

The remaining document is organized as follows: Section 2 describes a strategy for processing short NL queries and reports experiment results. Section 3 describes a strategy for data fusion and presents related experimental results. Section 4 describes a selective relevance feedback process and discusses related experimental results. Note that every experiment reported in these sections used the TREC4 ad hoc data and queries--our training materials for preparing for TREC5. Section 5 summarizes the training work. And finally, Section 6 comments on our TREC5 results.

2.0 Strategy for Processing Short NL Queries

The characteristics of short NL queries include 1) one sentence or one incomplete sentence, 2) lack of informative terms, and 3) lack of term frequency. The lack of query terms

leads to an increased level of query ambiguity, and the ambiguity in turn tends to lead to a deterioration of retrieval performance. On the other hand, the lack of term frequency information is detrimental to a statistical oriented retrieval system. The combination of these two factors could make the performance of an IR system deteriorate in the neighborhood of 30 percent [3]. Any strategy for processing short NL queries, to be effective, needs to reduce the level of ambiguity as well as to compensate for the loss of query term frequency.

One strategy proposed in [4] focuses on reducing the level of query ambiguity through a set of Boolean constraints. The strategy demands that searchers are capable of grouping the informative terms in a short NL query into subtopic sets (i.e., concepts or themes). The conjunction constraints in general require the retrieved documents to have most or all of the subtopic sets. The constraints can be implemented as a filtering mechanism on the top of a ranking system.

The Boolean constraints made a noticeable improvement to the short version of the TREC2 and TREC3 queries, but produced a mixed result on the short TREC4 queries. The reported improvement, however, may not be entirely attributed to the Boolean constraints due to the fact a number of the queries received additional terms during the manual construction of the queries. Any addition of quality terms to a short NL query could have a positive impact on its retrieval effectiveness. Also the requirement of manual construction of Boolean constraints makes this strategy less likely to be adopted as a real implementation.

Another strategy reported in [5] focuses on compensating for the loss of query term frequency in short NL queries. The strategy attempts to detect "significance" of each query term using several collection statistics such as average term frequency within a document and inverse term document frequency with a cut-off. The strategy made a significant improvement in retrieval effectiveness both in the original query runs as well as in the expanded query runs. Because the process is automatic, its chance for real application is great. It may be worthwhile to point out that the true average term frequency within a document is an expensive measurement, especially given a huge document collection; various estimations of the average, on the other hand, can be made with different confidence levels.

We have developed a different strategy for tackling both the problem of query ambiguity and the problem of lack of query term frequency. The assumption is that nouns and noun phrases, other than those popular ones, usually represent important concepts in short NL queries, and their inter-term distances in the queries approximate their conceptual distances [4, 6,7]. Taking TREC query 203, for example,

What is the economic impact of recycling tires?

the noun "economic" and the noun "recycling" are two words apart and their conceptual distance is "longer" than that between "recycling" and "tires". But it is debatable whether the conceptual distance between "economic" and "tires" is farther than that between "economic" and "recycling."

Taking TREC query 207 as an another example,

What are the prospects of the Quebec separatists achieving independence from the rest of Canada?

the noun “Quebec” and the noun “separatists” are closer to each other than “Quebec” and “Canada”, while “Quebec” and “Canada” are “equally” close to the noun “independence. As in many natural language analyses, one can find many supportive cases and one can also easily discover some negative situations. With an application in our mind, we are happy with a plausible assumption. From this assumption we hypothesize that:

- The further apart a pair of nouns appear in short NL queries, the further distinct the two represented concepts are from each other, and therefor the more desirable that the two concepts co-occur in retrieved documents (That is, a softer constraint than the Boolean conjunction [8]).
- And this 2-noun dependence expressed in a function of inter-term distance can improve the effectiveness of retrieving documents for short NL queries.

These hypotheses should not be confused with the theory of probabilistic term dependence usually expressed in the form of Bahadur Lazarsfeld expansion [9]. The hypotheses are only associated with the *initial* retrieval run before a relevance feedback run, and are not yet related to any estimation of probability parameters for a given set of relevant documents and a given set of non-relevant documents. In addition, the hypotheses only consider 2-noun dependence. The goal is to retrieve much more relevant documents at the top of a rank list so that a relevance feedback process, whether it is a probability based or the vector based, can be more effective. Another practical goal is to develop a handle with which one can have a certain degree of control on the quality of top-ranked documents.

To test the hypotheses, we selected the rank list from Cornell’s Lnu.ltu run as our benchmark [10], the best performer in handling the short TREC4 queries, and we also developed two ranking functions, one derived from Okapi BM25 [11] for favoring medium to long documents (M-L function for short) and another derived from Lnu.ltu [10] for favoring medium to short documents (M-S function). Some characteristics of these two functions are described in Table 1 and 2. Both functions can give a document a bonus score when they have a match to a 2-noun dependence specification. The bonus score is calculated using both the weights assigned to the two nouns and an inverse measure of their inter-term distance.

Function	Ave Ret Doc Length (byte)	Length Range
M-L	4,125	404 - 733,919
M-S	4,069	356 - 149,009
Lnu.ltu	4,664	388 - 252,888

TABLE 1. Length Characteristics of the Retrieved Documents

Function	Ret_Doc	Ret_Doc_Overlap	Ret_Rel_Doc	Ret_Rel_Doc_Overlap
M-L	49000	37,115	3634	3,407
M-S	49000		3665	
Lnu.ltu	49000	28,298	3709	3,126
M-S	49000		3665	

TABLE 2. Overlap of the Retrieved documents from the 3 functions

There are two types of test measures. The precision biased measures are the precision ratios at the six cut-off points among top 20 documents, namely, 1st, 3rd, 5th, 10th, 15th and 20th. These precision ratios allow us to closely observe the behavior changes of the two ranking functions. On the other hand, the recall biased measure is merely the number of relevant documents retrieved among the 49,000 retrieved documents for a TREC4 ad hoc run. This number has a strong correlation to the 11-point average precision measure but is more readable. Table 3 describes the test results. The positive percentages in parentheses are the ratios of improvement and the negative percentages are the ratios of deterioration, in comparison to the baseline. The punctuation “:” in the table headings separates the experimental conditions.

System	Lnu.ltu (baseline)	M-L	M-L;2-noun- dependence	M-S	M-S; 2-noun- dependence
Precisions					
1st	.53	.57 (8%)	.69 (30%)	.57 (8%)	.57 (8%)
3rd	.55	.58 (5%)	.63 (15%)	.52 (-6%)	.60 (9%)
5th	.51	.54 (6%)	.58 (14%)	.55 (8%)	.59 (16%)
10th	.46	.47 (2%)	.51 (11%)	.50 (9%)	.53 (15%)
15th	.44	.45 (2%)	.46 (5%)	.46 (5%)	.48 (9%)
20th	.41	.43 (5%)	.44 (7%)	.42 (2%)	.45 (10%)
Recall					
rel-doc-ret	3709	3634 (-2%)	3681 (-1%)	3665 (-1%)	3694 (0%)

TABLE 3. Query term dependence experiments

The results in Table 3 suggest that both M-L function and M-S function can perform at the same level of effectiveness as Lnu.ltu function does, if not at a better level in terms of precision. The introduction of 2-noun dependence to M-L function and M-S function does improve the effectiveness further. The improvements may not be statistically significant when compared to their base performance (M-L and M-S), but the precision improve-

ments over the baseline Lnu.ltu may be significant. The recall measure has been held steady in these experiments. The results support our hypotheses. In addition, we think that further improvement is achievable.

3.0 Strategy for Data Fusion

With the two performing ranking functions (M-L and M-S), we turned our attention to a data fusion strategy to further improve the retrieval effectiveness reported in Table 1. The data fusion approach is based on the phenomenon that different ranking functions tend to retrieve a different set of documents, and, more specifically, a different subset of relevant documents. A good fusion mechanism can retrieve a set of relevant documents that is larger than every individual subset of relevant documents from the individual ranking functions [1]. The most important feature is the ability to retrieve *different documents*. A list of ranking functions from the same family such as SMART [12], on the other hand, may share more commonalities than differences--making a fusion process less meaningful. This, in fact, was one of our motivations to develop the specialized ranking functions.

To emphasize the diversity of the ranking functions in our data fusion tests, we decided to add another function which ranks documents based entirely on their best local relevance scores [13, 14]. We defined a local text window as a list of query term occurrence(s) in a span of 250 words (including stopword positions) from a given query term. The number of text windows that a document has is determined by the number of occurrences of a given set of query terms. We were initially thinking of using the traditional cosine function [15] since it strongly favors short documents. But we actually used M-L function (M-L-LOC) because of the fact that the window has a fixed size.

We experimented with a few fusion methods and finally selected Logistic Regression [16]. The input to the regression analysis is a table consisting of eight fields, namely, binary relevance indicator, weighted publication indicator, the relevance score and the rank from M-L function, the relevance score and the rank from M-S function, and the relevance score and the rank from M-L-LOC function. The field weighted publication deserves additional description. This field can accommodate the information such as an estimated relevant document distribution among different publications or data sources (e.g., AP and FR), or user's preference of publications (e.g, Patent better than ZIFF). However, in preparing for the TREC5 runs we turned this field off by assigning a constant to it, since we did not have any knowledge about the data on Disk4 at that time.

The testing plan was to use the data on Disk2 as the training data and the data on Disk3 for testing. But due to time constraints, we used the both disks in training, and later applied the trained fusion function to the TREC5 ad hoc runs. As a result, we describe the training results in Table 4 and defer the discussion of the testing in the coming TREC5 section.

In Table 4, columns Lnu.ltu, M-L and M-S are copied from Table 3, and the baseline for comparisons is still Lnu.ltu. Column M-L-LOC covers the results from the local text window run. It is not surprising that this run has a serious deterioration from the baseline. The documents retrieved in this run are different from those from M-L and M-S, especially

those long documents with isolated relevant texts. The fusion training results are positive both in the precision measures and in the recall measure. The generalizability of this training will be discussed in the TREC5 section.

System	Lnu.ltu (baseline)	M-L	M-S	M-L-LOC	Fusion
Precisions					
1st	.53	.57 (8%)	.57 (8%)	.53 (0%)	.63 (19%)
3rd	.55	.58 (5%)	.52 (-6%)	.51 (-7%)	.63 (15%)
5th	.51	.54 (6%)	.55 (8%)	.48 (-6%)	.61 (20%)
10th	.46	.47 (2%)	.50 (9%)	.43 (-9%)	.52 (13%)
15th	.44	.45 (2%)	.46 (5%)	.40 (-9%)	.48 (9%)
20th	.41	.43 (5%)	.42 (2%)	.38 (-7%)	.46 (12%)
Recall					
rel_ret	3709	3634 (-2%)	3665 (-1%)	3401 (-9%)	3963 (7%)

TABLE 4. Data Fusion Training Results using the TREC4 Data

4.0 Strategy for Selective Relevance Feedback

Most systems involved in TRECs have adopted a 2-step approach in performing the ad hoc task: retrieve an initial set of, say, the top 20 documents for each test query, then feed the initial set into a relevance feedback process to retrieve the final rank list for that query. Usually these systems assume that every document in the initial set is relevant for the sake of automation. The flaw in this assumption is obvious--there are non-relevant documents in the initial set. If we could select relevant documents or remove non-relevant documents automatically, we would expect better performance from relevance feedback.

To this end, we deduced the following guidelines from the cluster hypothesis [3]:

- Both the relevant and the non-relevant documents in an initial set have a tendency towards grouping because they are all highly selected for a common query.
- The relevant documents, however, have a stronger tendency towards grouping than the non-relevant documents.
- In addition, the observation that every document in an initial set is clustered into one group usually indicates that the initial set contains many, if not all, relevant documents, and therefore a relevance feedback process will benefit most from this set.
- To generalize the above further, one large, stand-alone cluster after clustering has a large quantity of relevant documents. Detecting and extracting this cluster is easy.
- On the other hand, the observation that every document in an initial set is a cluster by itself (i.e., all singletons) probably suggests that there is no relevant document in the set, and therefore no relevance feedback should be initiated after clustering.

- Most singletons or orphan documents after clustering are non-relevant documents, and can be removed automatically.

Topic 207, perfect

Grp 1: 106458.r 106783.r 107790 110287.r 114271 117292.r 121838.r 123578.r 133508.r 146688.r 173399.r 269603.r 271890.r 278827.r 278926 279092.r 280452 306156.r 398792.r 289861.r

Topic 209, good

Grp 1: 000403 000434 000513.r 057655.r 066482.r 242256 243847 245128.r 245548 246061.r 246449 248076.r 248747.r 251392.r 286365

Grp 2: 044832 076895

Grp 3: 069711

Grp 4: 090430

Grp 5: 390461

Topic 232, bad

Grp 1: 015128

Grp 2: 025152

Grp 3: 048788

Grp 4: 079357

Grp 5: 143456

Grp 6: 153009

Grp 7: 241340

Grp 8: 260543

Grp 9: 269575

Grp 10: 290721.r

Grp 11: 292551

Grp 12: 304266

Grp 13: 312705

Grp 14: 315733.r

Grp 15: 345581 345879

Grp 16: 363641

Grp 17: 387106

Grp 18: 477459 486460

TABLE 5. Examples of Top 20 Document Clustering (TREC4)

Some examples based on the above guidelines are shown in Table 5, ranging from perfect to bad. The 6-digit document ids are from EUREKA and the small suffix letter “r” indicates a relevant document. The clustering process that generated the examples was a one-pass, flat grouping process. The similarity matrix used came from a variant of the cosine similarity function. A similarity threshold was selected for grouping the top ranked documents.

We assessed the reasonableness of the deduced guidelines. Table 6 contains a few sets of comparisons from the assessment. There are four similarity thresholds corresponding to the four singleton columns on the right. The thresholds are in an increasing sequence to make increasingly more but smaller clusters. The second threshold is also responsible for column "Largest Clust2." The baselines are the average precision of the 49 sets and the number of the sets that do not have any relevant document.

These two measures together provide a balanced view on the change of the 49 sets, since the increased precision leads to an increased number of sets that do not contain any relevant document. To minimize the number of sets consisting of all non-relevant documents as well as to maximize the density of relevant documents in the sets, we selected the strategy of removing the singletons from a clustering with a lower similarity threshold, instead of the strategy of selecting the largest clusters.

	Initial Sets	Largest Clust2	Singleton1	Singleton2	Singleton3	Singleton4
Ret_Docs	980	478	756	594	449	343
Ret_Rel_Docs	441	283	380	321	256	198
Ave_Precision	.45	.59	.52	.54	.57	.58
No_RelDoc_Sets	1	10	4	4	8	12

TABLE 6. 2 Different Approaches to Improving the Initial Set for Relevance Feedback

In Table 7 are the test results using the improved sets from "Singleton2". We used one of the relevance feedback functions in EUREKA. The function was assisted with and without the clustering processing. The symbol "RF" stands for relevance feedback, the symbol "clust" for clustering. The punctuation ";" separates the different ranking processes. The results show a positive contribution from the clustering analysis, but the contribution may not be significant in this test. The results are not conclusive.

System	Lnu.ltu	Lnu.ltu; RF	M-L;2-noun-dependence	M-L;2-noun-dependence;RF	M-L;2-noun-depdnce;clust;RF
Precisions					
1st	.53	.55 (4%)	.69 (30%)	.65 (23%)	.61 (15%)
3rd	.55	.53 (-4%)	.63 (15%)	.56 (2%)	.56 (2%)
5th	.51	.52 (2%)	.58 (14%)	.56 (10%)	.58 (14%)
10th	.46	.51 (11%)	.51 (11%)	.53 (15%)	.56 (22%)
15th	.44	.49 (11%)	.46 (5%)	.51 (16%)	.52 (18%)
20th	.41	.46 (12%)	.44 (7%)	.48 (17%)	.49 (20%)
Recall					
Rel_Ret	3709	4350 (17%)	3681 (-1%)	4219 (14%)	4335 (17%)

TABLE 7. Selective Relevance Feedback Experiment

5.0 Summary of our TREC4 Training

The strategy that applies 2-noun dependence in ranking seems help in improving the quality of top ranked documents. The improvement, however, is not dramatic. There is further potential for this indexing device. For example, by combining the query term selection strategies proposed in [6] we could make this device more precise. Furthermore, this device could be introduced into the probabilistic models in [17, 18] as part of the query networks.

The strategy that employs different or specialized ranking functions in data fusion seems help in improving overall retrieval effectiveness. But our fusion function trained with the TREC4 data is subject to the TREC5 test.

Being selective as to the top ranked documents in a process of relevance feedback seems help in improving retrieval effectiveness in general. The test results reported, however, stopped short of the 10% improvement margin, the margin usually implies a statistically significance. Perhaps with a better tuned clustering algorithm, the 10% improvement is attainable.

6.0 TREC5 Results

We submitted two runs, LNaDesc1 and LNaDesc2, for the required short query test (i.e., using the description section only). LNaDesc1 came from the trained TREC4 fusion function. The fusion combined the 3 rank lists from M-L function, M-S function, and M-L-LOC function. M-L function and M-S function were assisted with the 2-noun dependence device as well as the selective relevance feedback. The rank list from M-L function, also assisted with the 2-noun dependence device and the selective relevance feedback, was submitted individually as LNaDesc2, due to the uncertainty surrounding the new data on Disk4 and over the generalizability of the fusion function. Submitting the M-L rank list rather than the M-S list was due to the concern that its parent function Lnu.ltu might have been over fitted to the TREC4 materials.

The fact that LNaDesc2 performed better than LNaDesc1 in TREC5 indicates that the TREC4 fusion function suffered from over-fitting to the TREC4 data. But the performance deterioration is marginal. We will look into our TREC5 results after the conference.

The other two ad hoc submissions, LNmFull1 and LNmFull2, were incomplete runs. The original plan for the two manual runs was to use a statistical thesaurus [3] compiled from the data on Disk 3 and 4 to construct the test queries. Unfortunately the statistical thesaurus did not get compiled on time for the test runs, forcing us to abandon that plan.

Instead, we re-ran the processes for generating LNaDesc1 and LNaDesc2 with the addition of the manually edited full queries. For these two runs, we did not use the 2-noun dependence device because it might get “disoriented” in long queries. Finally, we did not perform the clustering of the top ranked documents for these two runs. Our motivation for submitting these runs is to establish baselines for our future research. For example, we

have been thinking of the process in which a human subject is given an opportunity to manually select one or more document clusters for manual relevance feedback [19]. We may just do that for the final conference paper.

Acknowledgments: The authors of this note would like to thank Daniel Pliske, David Miller and Vladimir Nayfeld for their reviewing work.

REFERENCES:

- [1] Belkin, N. J., Kantor, P., Cool, C. and Quatrain, R. "Combining evidence for information retrieval," The Second Text REtrieval Conference (TREC2), NIST Special Publication 500-215, pp.35-44, edited by D. Harman, 1994.
- [2] van Rijsbergen, C. J. Information Retrieval. Butterworths, London, 1979.
- [3] Lu, X. A. and Keefer, R. B. "Query expansion/reduction and its impact on retrieval effectiveness," The Third Text REtrieval Conference (TREC3), NIST Special Publication 500-225, pp.231-239, edited by D. Harman, 1995.
- [4] Hearst, M. A. "Improving full-text precision on short queries using simple constraints," Proceedings of 5th Annual Symposium on Document Analysis and Information Retrieval, pp.217-232, Las Vegas, Nevada, 1996.
- [5] Kwok, K. L. "A new method of weighting query terms for ad-hoc retrieval," In Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.187-196, Zurich, Switzerland, 1996.
- [6] Keen, E. M. "The use of term position devices in ranked output experiments," Journal of Documentation, Vol.47, pp.1-22, 1991.
- [7] Searle, J. R. Speech Acts. Cambridge University Press, London, 1984.
- [8] Salton, G., Fox, E. A. and Wu, H. "Extended Boolean information retrieval," Communication of the ACM, Vol.26, pp.1022-1036, 1983.
- [9] Yu, C. T., Buckley, C., Lam, K. and Salton, G. "A generalized term dependence model in information retrieval," Information Technology: Research and Development, Vol.2, pp.129-154, 1983.
- [10] Singhal, A., Salton, G. Mitra, M. and Buckley, C. "Document length normalization," Information Processing & Management, Vol.32, pp.619-633, 1996.
- [11] Robertson, S. E., Walker, S. Jones, S., Hancock-Beaulieu, M. M. and Gatford, M. "Okapi at TREC3," The Third Text REtrieval Conference (TREC3), NIST SPecial Publication 500-225, pp.109-126, edited by D. Harman, 1995.

- [12] Fox, E. A., Koushik, M. P., Shaw, J. Modlin, R. and Rao, D. "Combining evidence from multiple searches," The First Text REtrieval Conference (TREC1), NIST Special Publication 500-207, pp.319-328, edited by D. Harman, 1993.
- [13] Buckley, C., Salton, G. and Allan, J. "Automatic retrieval with locality information using SMART," The First Text REtrieval Conference (TREC1), NIST Special Publication 500-207, pp.59-72, edited by D. Harman, 1993.
- [14] Callan, J. P. "Passage-level evidence in document retrieval," In Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.302-310, Dublin, Ireland, 1994.
- [15] Salton, G. and McGill, M. J. Introduction to Modern Information Retrieval, McGraw-Hill Book Co., NY, 1983.
- [16] Cooper, W. S., Chen, A. and Gey, F. C. "Full text retrieval based on probabilistic equations with coefficients fitted by Logistic Regression," The Second Text REtrieval Conference (TREC2), NIST Special Publication 500-215, pp.57-66, edited by D. Harman, 1994.
- [17] Turtle, H. R. and Croft, W. B. "Evaluation of an inference network-based retrieval model," ACM Transactions on Information Systems, Vol.9, pp.187-222, 1991.
- [18] Kwok, K. L. "A network approach to probabilistic information retrieval," ACM Transactions on Office Information Systems, Vol.13, pp.325-353, 1995.
- [19] Hearst, M. A. and Pedersen, J. O. "Reexamining the cluster hypothesis: Scatter/Gather on retrieval results," In Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.76-84, Zurich, Switzerland, 1996.

InTEXT Automatic Query Enhancement in TREC-5

by

**Richard Jones, Mark Burnett, and Lewis Pape
InTEXT Systems,
P.O Box 310 Deakin West
ACT 2600 Australia**

100032.1075@compuserve.com

A. Background

InTEXT Systems is a subsidiary of the CP Software Group, whose headquarters are in Folsom, California. InTEXT is a leader in the provision of advanced tools and end-user products that can best be described as doing 'smart things with text'. The InTEXT Research and Development group, based in Canberra Australia, has been in existence for 11 years. We develop leading edge technology in areas including text retrieval, filtering, indexing, summarisation and thematic clustering, using the InTEXT Heuristic Learning Architecture.

The InTEXT Systems R and D group has participated in two previous TRECs. We took part in TREC-4 and also in TREC-1, when the team formed the Centre for Electronic Document Research (CEDR), in conjunction with CITRI.

B. Objectives of the Experiment

The experiments INTXA, and INTXM described in this paper were carried out in the filtering stream of TREC-5. The principal objective of these experiments was to determine the effectiveness of the InTEXT Precision indexing tool to automatically improve queries. The experiments were run with the InTEXT Object Router tool-kit, and a second objective was to determine the effectiveness of Object Router in the TREC filtering environment.

In particular we wanted to see if InTEXT Precision could be used to take an initial query, manually created with minimal effort and transform it into an improved query, taking automatically selected words and phrases from a group of documents that were deemed relevant to the original query

The motivation behind the experiment was that in a filtering environment, queries are long-lived objects and so the return on investment in improving them will be greater than for an ad-hoc query. However, end users do not enjoy the intricacies of query building. In addition, a time specific cohort of documents relevant to the query will share a number of concepts other than those that caused them to be selected. These terms can be useful in

improving a query. However, some terms will be ephemeral (eg references to the Exxon Valdez in connection with environmental disasters), and a different document cohort from a different time will have an overlapping, but not identical set of concepts that distinguish them. Therefore a query needs to be modified with time, but if users are reluctant to create a good initial query, they will be doubly reluctant to keep it up to date.

C. The Experiment

The initial 50 queries were built manually using the information contained in the original TREC queries as a base (except in those taken from TREC-4, where we used our own queries). Minimal effort was used in building the queries. This set formed the experiment INTXM.

All the training documents judged relevant to the queries in earlier TREC experiments were extracted and these were run through all the INTXM queries in InTEXT Object Router. For each query the 15 documents judged most relevant by Object Router were selected. Those paragraphs containing search terms were extracted from each document and concatenated to form a composite document.

These 50 composite documents were run through InTEXT Precision to extract a set of key words, ordered by their automatically generated weights. These measure their contribution to the information content of the document.

The keywords were then used to create a second set of queries, following the rules defined in Appendix 1. Note that some existing terms were removed or modified by the rules, as well as new ones being added. For an example of a pair of queries for the same topic, see Appendix 2. These queries formed the experimental set INTXA.

Both sets of queries were dispatched to the TREC organisers. The test documents were run through the 100 queries from the combined experiment sets, and the selections made for evaluation under the three utility functions defined in the experiment.

Object Router derives a relevance score for each document, based on the criteria outlined below. The method chosen for document selection was to take a cut-off at a percentage of the highest relevance score achieved for a given query. The percentages selected were: 66%, 33% and 16% from Precision Oriented to Recall oriented.

D. The InTEXT Tool-kits

InTEXT Object Router provides a measurement of the relevance of a document stream to a number of queries. The queries or filters may either be expressed in plain English, or in a structured form, as in Appendix 2, together with a threshold score in the range 1-100 defining the relevance that a document must reach to be passed to the filter owner. Relevance is measured by a set of heuristics, which include presence of the number of terms, and their distribution through the document, with more value being assigned to

closely grouped terms. The software, which was used in an earlier version in TREC-1 has two interesting features:

- the queries are indexed since they are the long lived objects, but the documents are not, as they are ephemeral. As a result the Object Router can efficiently support many active filters, in excess of 100,000;
- the query terms are automatically weighted, depending on their ability in the context of the particular query to act as a good discriminator. The weighting is carried out dynamically on the document stream, so the term weights did vary during the experimental runs.

InTEXT Precision was described in TREC-4. It analyses document content, using techniques analogous to skim reading to identify, rank and categorise keywords and phrases, referred to as Keys. Queries were automatically generated from the list of Keys, depending on their rank and type (for example, proper noun, acronym etc).

Precision also generates 'skinny documents' to support full text indexing whilst effectively generating a stop word list for each document and reducing the number of words to be indexed by a factor of 10. This was the basis of the TREC-4 experiment

E. Results of the Experiments

The basic question we were interested in answering in the experiment was whether INTXA earned more money (or lost less) than INTXM.

The mean results per query using the Pool utility calculation were as follows:

	Precision-Oriented	Balanced	Recall-Oriented
INTXA	-11	-32	-48
INTXM	-281	-451	-405

So on the base figure from the utility function, the modified queries performed considerably better than the unmodified one, even though neither were brilliant.

Turning to precision and recall across the 45 queries with relevant documents, the official results are:

INTXM	Precision	Balanced	Recall
Total Retrieved	5512	27520	30002
Total Relevant	5808	5808	5808
Relevant Retrieved	611	2322	2438

INTXA	Precision	Balanced	Recall
Total Retrieved	393	3126	9567
Total Relevant	5808	5808	5808
Relevant Retrieved	149	757	1787

INTXM	Precision	Balanced	Recall
Mean Precision	0.3308	0.1679	0.1654
Mean Recall	0.1068	0.4212	0.4338

INTXA	Precision	Balanced	Recall
Mean Precision	0.3331	0.2326	0.1913
Mean Recall	0.0793	0.1894	0.3295

On these statistics, the improvement is not at all clear-cut. though see comment below on improvment on precision figures from above data.

F. Comments

The results are encouraging in the following regard:

- the automatically generated terms looked sensible - see Appendix 2 for an example;
- the automatically generated queries would be particularly useful in an environment demanding high precision. For example, in the Precision-Oriented automatic run 149 out of 393 documents routed were relevant, compared to 611 out of 5512 for the manual queries.

Why the results were not better:

- the training documents from which the modified queries were built were very different from those used in the experiment, in time and in source. The mechanism will work better in an environment where there is some continuity across the data.
- Some of the rules for query modification were too aggressive, in replacing existing terms sometimes with terms that were ephemeral;
- it is clear that the cut-off point decisions were not optimal.

G. Conclusion

We are satisfied that the approach has the potential to gives worth-while results, but that work is needed to improve strategies as described in the comments above.

Appendix 1 - Query Generation/ modification Procedure

A query is made up of terms, approximating to a concept. A term is made up of a number of tokens, which are alternatives. A token may be a word, a phrase, or words collocated in paragraph. In the Object Router, a query is limited to 20 terms. See specific query syntax in Appendix 2.

The original query is modified by the addition of new tokens either into existing terms, or as new terms, and by the removal of original tokens using the rules described below. New tokens are selected in turn from the Keys generated by Precision, starting with the highest scored. For comparison purposes, all Keys and tokens are truncated with the same stemming algorithm.

When the Keys are combined, all proper noun terms with only one token are combined into one term, and the other single token terms are also combined. Then if necessary, tokens are successively removed starting with the lowest ranked by Precision, until the limit is reached.

Rules for incorporating Keys

1. Proper Nouns

- Add as phrase, not paragraph collocation
- add as a new term, unless linked to another token by a shared word
- Accept 1, 2, or 3 word phrases
- match acronyms to proper compound nouns, using InTEXT heuristics

2. Non Proper Nouns

- Add phrase as 'same paragraph' collocation.
- Do not add single words, phrases counted after elimination of noise words
- Allow insertion of 3 word phrases if in top 10 Keys.
- Do not replace two word phrase by three word phrase
- If two 3 word phrases have two words in common, replace by the two word sub-phrase

3. Token addition/ replacement/ deletion

- If A B exists, add any two word phrase containing A and B as extra token
- Delete all tokens not in the keyword list, unless its component words all exist
- If A or B exists as a single word token and A B occurs above A or B in the keyword list then replace by A B. Else add as new term.
- If term choice in token addition, add to shortest term
- do not add tokens to mandatory terms

4. Term addition

- Add 2 term phrases or proper nouns with no current matches as new terms.

Appendix 2 - Example Queries

+ Mandatory term

// same paragraph

* stemmed

INTXM Query 142

1. +grain, animal feed*, corn, rice, wheat
2. export subsid*, export restitution, import quota*, farm trade barrier*
3. price support*, farm subsid*, agricultural support*
4. self sufficiency, embargo, import dependence, high quality food supply, farm interests
5. foreign trade, trade negotiations, agricultural policy
6. Common Agricultural Policy, CAP, Ministry//Agriculture, Zenchu, Liberal Democratic Party, LDP, Export Enhancement, food importing

InTXA Query 142

1. +grain, corn, rice, wheat
2. export subsid*, export restitution, import quota*, export enhance*
3. price support*, farm subsid*, agricultur* support*, low*//price*
4. self sufficiency, embargo, import dependence, farm interests, American//farm*, Rice//farm*, farm//product*
5. foreign trade, trade negotiations, agricultural policy
6. Ministry//Agriculture
7. food import*, import*//quota*
8. import*//rice, rice//miller*, rice//market*
9. export subsidy, wheat//export*
10. Japan*//rice, Japan*//beef

INTXM Query 189

1. +murder*, homicide*, second degree
2. motive*, reasons, jealous*, relation*, partner, family, domestic

INTXA Query 189

1. +murder*, second degree
2. motive*, family
3. capital//murder, commit//murder, first-degree//murder, second-degree//murder, open//murder, accused//murder//conspiracy, murder//plot, murder//charges
4. Murder Conspiracy Trial, Appeals Court
5. famil*//member*, famil*//reunion, famil*//honor, family//problem
6. death//penalty, alleged//plot, racial//motive, defense//attorney

Experiments on Routing, Filtering and Chinese Text Retrieval in TREC-5

Chong-Wah Ngo ¹

Kok F. Lai ²

Information Technology Institute
11 Science Park Road, Singapore 117685.

Abstract

We describes our experiments in the routing, filtering and Chinese text retrieval. We based our routing and filtering experiments on our discriminant project algorithm. The algorithm sequentially constructs a series of orthogonal axis from the training documents using the Gram-Schmidt procedure. It then rotates the resulting subspace using principal component analysis so that the axis are ordered by their importance. For Chinese text retrieval, we experimented both with an automatic method and a manual method. For the automatic method, we use all phrases in the description field and compute the aggregate scores using the simple *tf.idf* formula. We then manually construct boolean phrase queries which are thought to improve the results.

1 Introduction

This paper describes the routing, filtering and Chinese text retrieval experiments performed by the Information Technology Institute for TREC-5.

We based our routing and filtering experiments on our *discriminant project algorithm* as described in TREC-4 [1]. The algorithm sequentially constructs a series of orthogonal axis from the training documents using the Gram-Schmidt procedure. It then rotates the resulting subspace using principal component analysis so that the axis are ordered by their importance. One can then discard insignificant axis to obtain compact and efficient representation of relevance information.

For the routing task, we experimented with a two-phase training algorithm. In the first phase, we used the centroid of all relevant documents to produce an initial ranking for all training documents. We then select the top X % of non-relevant documents from the initial ranking, and then supply them as negative examples into the discriminant projection algorithm. We constructed the two routing queries, *itidp1* and *itidp2*, using $X = 10$ % and $X = 30$ % respectively.

For the filtering task, we used *itidp1* as the filtering query and attempted to find the best threshold to maximise the utility measures. The training documents are first divided equally into two sets. We first construct the routing queries from the first set using the discriminant projection algorithm. We then use these queries to obtain the scores for all documents in the second set. For each query, we find the respective thresholds that best optimise the utility measures. We then apply these thresholds,

¹Currently with Computer Science, Hong Kong University of Science and Technology

²Please direct all enquiries and correspondence to Kok F. Lai, Information Technology Institute, 11 Science Park Road, Singapore 117685. Email address : kflai@iti.gov.sg.

together with routing query constructed from the entire training set, to filter new documents in the test set.

For Chinese text retrieval, we experimented both with an automatic method and a manual method. For the automatic method, we use all *phrases* in the description field and compute the aggregate scores using the simple *tf.idf* formula. We then examine the results and manually construct boolean phrase queries which are thought to improve the results.

The following sections explain in more details the experiments which were performed.

2 The Discriminant Projection Algorithm

Denote a document as follows :

$$\mathbf{d} = [f_1, f_2, \dots, f_n]^T \quad (1)$$

where f_j is a function of term frequency for term j . n is the total number of unique indexed terms, and is typically very large. We shall assume that \mathbf{d} is normalized, i.e., $|\mathbf{d}| = 1$, in subsequent discussions.

Denote the collection of training documents as $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_m]$. For a typical routing task, \mathbf{D} consists of two separate groups : \mathbf{D}_r and \mathbf{D}_x . \mathbf{D}_r contains the set of m_r relevant documents while \mathbf{D}_x contains the set of m_x non-relevant documents.

Note that the total number of training documents, $m = m_r + m_x$. Without loss of generality, let

$$\mathbf{D} = [\mathbf{D}_r : \mathbf{D}_x] \quad (2)$$

We want to find a *routing query* \mathbf{q} , such that $\mathbf{D}_r^T \mathbf{q} = \mathbf{1}$ and $\mathbf{D}_x^T \mathbf{q} = \mathbf{0}$. Thus, we write

$$[\mathbf{D}_r : \mathbf{D}_x]^T \mathbf{q} = [\mathbf{1} : \mathbf{0}]^T \quad (3)$$

Using the method of least squares, it can be shown [1] that solution to (3) is given by

$$\mathbf{q} = \mathbf{R}^{-1} \sum_{i=1}^{m_r} \mathbf{d}_{r_i} \quad (4)$$

The solution involves the inversion of a very large $n \times n$ correlation matrix, $\mathbf{R} = \mathbf{D}\mathbf{D}^T$. Unfortunately, unlike standard pattern recognition problems, document routing tasks are peculiar as they typically create very sparse, or *empty* feature spaces. That is, the number of training documents, m , is usually very small compared to the number of dimensions, n . Thus, \mathbf{R} is usually not invertible.

Consequently, it is necessary to project the raw document space into a lower dimensional space where each axis possesses higher representational richness. We can project the raw document space \mathbf{D} into a lower dimensional space \mathbf{V} as follows:

$$\mathbf{V} = \mathbf{U}^T \mathbf{D} \quad (5)$$

where $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k]$ are the first k orthonormal basis that account for most information in the raw space. The solution for routing query \mathbf{q} is then given by (see [1]) :

$$\mathbf{q} = \mathbf{U}\mathbf{U}^{-1} \sum_{i=1}^{m_r} \mathbf{U}^T \mathbf{d}_{r_i} \quad (6)$$

where

$$\Lambda = \mathbf{V}\mathbf{V}^T = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_k) \quad (7)$$

is a diagonal matrix containing the eigenvalues of \mathbf{R} in descending order.

Now, \mathbf{U} and Λ can be computed by performing principal component analysis on matrix \mathbf{R} [3]. This method is equivalent to singular value decomposition³ (SVD) employed by Latent Semantic Indexing [4]. However, the order of complexity of the standard SVD algorithm [5] is $O(n^2 + k^3)$. Since frequently $n \gg m$, it will be advantageous to find an alternative method that exploits this characteristics.

The next two subsections describe the discriminant projection algorithm, which uses only the m documents with relevance information to compute the optimum projection. It first constructs a subspace based on the training documents using the Gram-Schmidt procedure. The dimension of this subspace will be at most m . It then performs principal component analysis on this subspace to extract orthonormal basis which are ordered by their importance in capturing the subspace information. By specifying the desirable representation richness, only $k < m$ orthonormal basis needs to be extracted.

2.1 Gram-Schmidt Procedure

The procedure makes use of training documents $\mathbf{d} \in \mathbf{D}$. We begin the procedure with the initialization

$$\mathbf{y}_1 = \mathbf{d}_1 \quad (8)$$

We then construct \mathbf{y}_2 as follows :

$$\begin{aligned} \mathbf{x}_2 &= \mathbf{d}_2 - \mathbf{d}_2^T \mathbf{y}_1 \mathbf{y}_1 \\ \mathbf{y}_2 &= \frac{\mathbf{x}_2}{|\mathbf{x}_2|} \end{aligned} \quad (9)$$

The vector $\mathbf{y}_2 = \mathbf{0}$ if and only if $\mathbf{d}_2 = \alpha \mathbf{d}_1$, meaning that \mathbf{d}_2 is linearly dependent upon \mathbf{d}_1 . Assume that $\mathbf{y}_2 \neq \mathbf{0}$, then \mathbf{y}_1 and \mathbf{y}_2 are linearly independent and orthogonal :

$$\mathbf{y}_2^T \mathbf{y}_1 = 0 \quad (10)$$

We continue this procedure, generating each new vector \mathbf{y}_j as follows :

$$\begin{aligned} \mathbf{x}_j &= \mathbf{d}_j - \sum_{i=1}^{j-1} (\mathbf{d}_j^T \mathbf{y}_i) \mathbf{y}_i \\ \mathbf{y}_j &= \frac{\mathbf{x}_j}{|\mathbf{x}_j|} \end{aligned} \quad (11)$$

This procedure maps the vectors $(\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_m)$ into vectors $(\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_m)$ vectors. In the latter set, there will be $r \leq M$ nonzero vectors that are linearly independent which are retained. The discarded vectors are linearly dependent on the r linearly independent vectors and do not carry additional information on the training documents.

Note that the complexity of this algorithm is $O(nr^2)$ and is linear with respect to the number of terms n . In the worst case, the complexity will be $O(nm^2)$.

³Please refer to [3] for the relationships between singular values and eigenvalues.

2.2 Eigen-Decomposition of the Resultant Subspace

Without loss of generality, we write the resultant nonzero vectors as follows : $(\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_r)$. We can obtain the $r \times r$ correlation matrix \mathbf{G} in this space by simply computing the projection on each training documents on the \mathbf{y} vectors :

$$\mathbf{G}(p, q) = \sum_{i=1}^m \mathbf{d}_i^T \mathbf{y}_p \cdot \mathbf{d}_i^T \mathbf{y}_q \quad (12)$$

We can then apply principal component analysis to extract eigenvectors of this r -dimensional subspace. In our implementation, we use the Hotelling's Iterative Procedure [7] to solve the eigen-decomposition problem. The procedure iteratively obtain eigenvectors $\mathbf{w}_1, \mathbf{w}_2, \dots$ that corresponds to eigenvalues $\lambda_1, \lambda_2, \dots$ in decending order. One can simply terminate the procedure if a pre-specified threshold t has been exceeded :

$$\sum_{i=1}^k \lambda_i / \text{tr}(\mathbf{G}) \geq t \quad (13)$$

$\text{tr}(\mathbf{G})$ is the trace of the matrix \mathbf{G} that denotes the *total* variance available in the training documents. For example, if $t = 0.75$, then the first k eigenvectors account for more than 75% of the information in the training documents.

In the Hotelling's procedure, the iteration of each eigenvectors typically take only a few steps to converge. Since the algorithm operates on \mathbf{G} , its complexity is $O(m^2)$ in the worst case.

Finally, the orthonormal basis \mathbf{u} can be obtained via

$$\mathbf{u}_i = \sum_{j=1}^k w_{ij} \mathbf{y}_j \quad (14)$$

where w_{ij} are the coefficients from eigenvectors \mathbf{w}_i . Each \mathbf{u}_i , in diminishing importance, captures an orthogonal axis containing the semantics of the training documents. Thus \mathbf{u}_1 is a linear combination of terms containing the highest information contents in the training documents. \mathbf{u}_1 is orthogonal to \mathbf{u}_2 i.e., $\mathbf{u}_1^T \mathbf{u}_2 = 0$: it captures information contents missed by \mathbf{u}_1 , and so on.

2.3 Applying Discriminant Projection Sequentially

Now, suppose that we have applied semantic projection on a set of $m - 1$ documents to obtain a set of k orthogonal basis in $\mathbf{U}_{m-1} = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k]$. Correspondingly, we have a $k \times k$ correlation matrix Λ_{m-1} which contains the eigenvalues in its diagonal. We wish now to expand our semantic projection to encode information of a new document \mathbf{d}_m .

To achieve this purpose, we simply restart the Gram-Schmidt procedure by initializing

$$\mathbf{y}_i = \mathbf{u}_i \quad (15)$$

for $i = 1, 2, \dots, k$. We then obtain the new Gram-Schmidt vector \mathbf{y}_{k+1} using (11) and the corresponding correlation matrix \mathbf{G}_m using (12). With a little book-keeping, it can easily be shown that

$$\mathbf{G}_m = \begin{bmatrix} \Lambda_{m-1} & \mathbf{U}_{m-1}^T \mathbf{y}_{k+1} \\ \mathbf{y}_{k+1}^T \mathbf{U}_{m-1} & \epsilon \end{bmatrix} \quad (16)$$

where $\epsilon = 1 - \mathbf{y}_{k+1}^T \mathbf{U}_{m-1} \mathbf{U}_{m-1}^T \mathbf{y}_{k+1}$ is a measure of *residual* projection. We can now obtain a new set of orthogonal basis \mathbf{U}_m by performing principal component analysis on \mathbf{G}_m .

3 Routing Experiments

We experimented only with documents that appear on the relevant list of at least one query. All other documents that do not belong to this group are ignored. Consequently our training documents are drawn only from approximately 18,000 documents, rather than from the entire TREC collection.

Each document is automatically parsed and index terms representing the document content are retrieved. A simple token recognizer written in flex is used to identify words within the document structure that contributes to the content. Most words are indexed except very common words, maintained in a stopwords list, and numbers are removed. The remaining words are then stemmed using the Porter's stemming algorithm. The document indexing process next creates the inverted files using Faircom's c-tree Plus file management product. Currently, without compression, the overhead for the indices amounts to about 80% of the original database size.

Using the training documents, we compute the idf weights for each term, *i.e.* $idf_i = \log(N/n_i)$. Weights of terms seen only in test documents but not in training documents are set to 1.

We construct the initial routing query \mathbf{q}_i by using all relevant documents. It can be shown that $\mathbf{q}_i = \mathbf{m}$, the mean (centroid) of the relevant documents. Using \mathbf{q}_i , we ranked all training documents \mathbf{d} by computing their projection score $\mathbf{q}_i^T \mathbf{d}$.

From the initial ranking, we select the top X % of the non-relevant documents, and compute the new routing query using the sequential formula. For *itidp1*, we set X to 10 %, *i.e.*, we use $0.1 * m_r$, where m_r is the number of relevant documents. For *itidp2*, we set X to 30 %. The results of the experiments are shown in Figure 1.

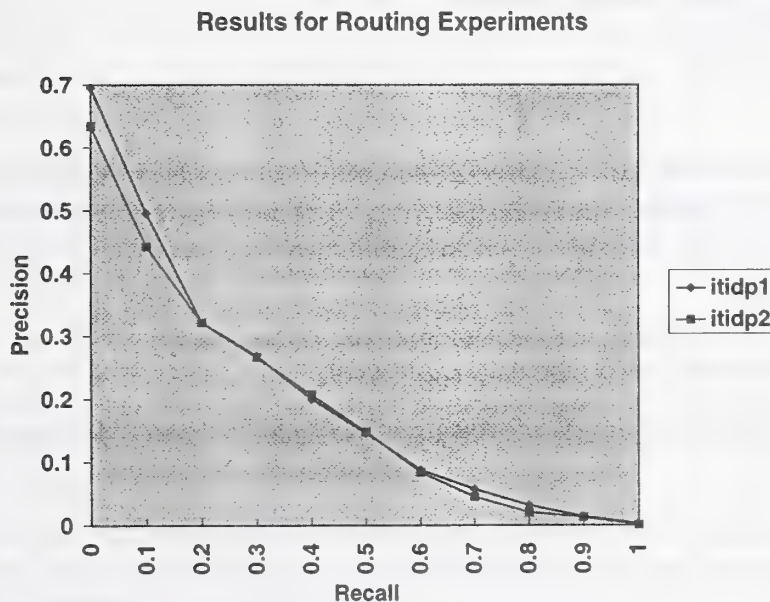


Figure 1: Results for Routing Experiments

From the experience gathered through our previous experiments, we have expected that *itidp2* to be a high precision run, out-performing *itidp1* in the high precision region. This is because more non-relevant documents were used in constructing *itidp2*. However, the results obtained show that *itidp1* consistently out-performs *itidp2* at all recall levels.

In our routing experiments, we have not used the original query topics, but instead relied completely on the training documents. On hindsight, this is probably a mistake. In our post-mortem, we have found that we can significantly improve the results by assigning heavy weights to the original query topics.

4 Filtering Experiments

We used *itidp1* as the filtering query and attempted to find the best threshold to maximise the utility measures. We first divide the training documents a equally into two sets, and construct the routing queries from the first set using the discriminant projection algorithm. We then use these queries to obtain the scores for all documents in the second set. For each query, we find the respective thresholds that best optimise the utility measures. We then apply these thresholds, together with routing query constructed from the entire training set, to filter new documents in the test set.

5 Chinese Text Retrieval

We experimented both with an automatic method and a manual method.

For the automatic method (*itcn1*), we use all *phrases* in the description field and compute the aggregate scores using the following formula :

$$s_k(\mathbf{d}_i) = \frac{\sum_{j=1}^Q (IDF_j * freq_{ij})}{C * \sqrt{\mathbf{d}^T \mathbf{d}}} \quad (17)$$

where Q is the number of matching phrases between document i and query k , IDF_j is the IDF weight for term j in the entire collection, and $freq_{ij}$ is the frequency of phrase j in document i . C is a constant which is selected to avoid heavily penalising long documents, and is set to 1000 in our experiment.

For the manual method *itcn2*, we construct boolean phrase queries which are thought to improve the results. The queries used are shown in Figure 5.

The boolean constructs are fuzzy, *i.e.*, let $s_x(\mathbf{d}_i)$ and $s_y(\mathbf{d}_i)$ are scores obtained from two phrase queries x and y , then

$$\begin{aligned} x \text{ AND } y &\rightarrow \min(s_x, s_y) \\ x \text{ OR } y &\rightarrow \max(s_x, s_y) \\ x \text{ NOT } y &\rightarrow \min(s_x, 1 - s_y) \\ x | y &\rightarrow s_x + s_y \end{aligned} \quad (18)$$

The results for *itcn1* and *itcn2* are shown in Figure 3. It can be seen that adding boolean constructs significantly improve the results.

- | | |
|---|--|
| 1 最惠国待遇 and 人权 and (分离 or 脱钩) | 15 海地 and (联合国 or 美国 or 多国部队 or 维和部队) and 民主 |
| 2 中国 and 台湾 and 一国两制 | 16 伊拉克 and 经济制裁 |
| 3 核电站 大亚湾 泰山 安全 | 17 亚太经济合作组织 中国 关贸总协 世界贸易组织 |
| 4 油田 天然气 油气 储量 油质 | 18 以色列 巴勒斯坦 中东 和平会议 |
| 5 (知识产权法 or 商标法 or 著作权法 or 专利法) and 保护 not 贸易制裁 | 19 中国 希望工程 文化程度 教育 |
| 6 中国 and (世界贸易 and (组织 or 体系) or 关贸总协 or 市场准入 or 多边贸易) and (成员 or 加入 or 支持) | 20 越南 and 失踪 and (美军 or 美国 and 军人) |
| 7 (南沙 or 东沙 or 西沙) and (主权 or 争端 or 建议 or 资源) | 21 (香港 or 特别行政区) and 彭定康 |
| 8 日本 and 地震 | 22 疟疾 and (死亡 or 感染 or 病例) |
| 9 (中国 or 我国) and (毒品 or 可卡因 or 大麻 or 海洛因) | 23 苏联 and (调停 or 和平建议) and (海湾战争 or 伊拉克) |
| 10 (新疆 or 维吾尔) and (边境贸易 or 边贸) | 24 (波黑 or 波斯尼亚 or 黑塞哥维那) and 武器禁运 and (取消 or 解除) |
| 11 (波斯尼亚 or 前南斯拉夫 or 巴尔干) and 联合国 and (维和 or 维持和平) | 25 熊猫 and (生态 or 环境 or 保护 or 灭绝 or 繁殖) |
| 12 妇女大会 and (妇女问题 or 妇女地位) | 26 (中国 or 蒙古 or 东北) and 森林 and (火灾 or 大火 or 防火) |
| 13 中国 and (奥运 or 世界运动大会 or 奥林匹克) and 申办 | 27 (中国 or 我国) and 机器人 |
| 14 (中国 or 云南 or 我国) and (艾滋病 or 艾滋病 or HIV) | 28 (中国 or 我国) and 移动电话 and (数字 or 蜂窝式 or 网络 or 自动漫游) |

Figure 2: Manual Queries used in Chinese Experiments

Through these experiments, a major difference between Chinese and English text retrieval was discovered. Typically in English text retrieval, the number of terms M , is significantly greater than the number of documents, N , i.e. $M \gg N$). This is not so in Chinese text retrieval. There are less than 10,000 Chinese characters encoded in the GB codeset, and this number remains constant even when N keeps increasing.

This is an important characteristics which must be taken into account when designing a Chinese text retrieval system. In our existing system which was originally designed for the English language, phrase retrieval is performed by first retrieving the position lists of the constituent terms, followed by proximity check. Since $M \ll N$ for large N , each term is likely to occur in many documents, making proximity verification very expensive to perform.

This problem can be alleviated by performing word segmentation or by using n-gram processings. Both methods increase M and in some cases eliminate the needs to perform proximity verification. This approach of retrofitting a text retrieval system from English language to Chinese language appears to be very popular among the Chinese track participants.

We intend to deviate significantly from this paradigm in our next participations. We believe that a different paradigm, designed by taking into considerations the unique characteristics of Chinese language, should be proposed and tested.

Results for Chinese Retrieval Experiments

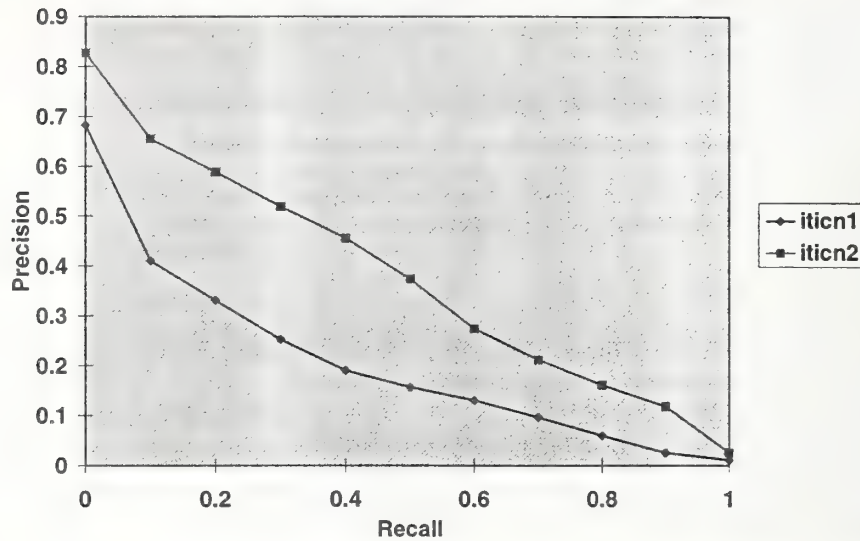


Figure 3: Results for Chinese Retrieval Experiments

6 Conclusion

We have described our experiments in the routing, filtering and Chinese text retrieval experiments. The following are our observations :

- For routing experiments, using more non-relevant documents appear to have hurt performance. It is probably a mistake to use only the training documents but ignore the original query topics.
- In our Chinese text retrieval, using less query phrases operated by Boolean conjuncts increased performance significantly.

References

- [1] Kok F. Lai, Vincent Lee & Jeremy Chew, "Document Routing by Discriminant Projection," *Overview of the Fourth Text Retrieval Conference (TREC-4)*, 1995, to appear.
- [2] D. Harman, "The DARPA TIPSTER Project," *SIGIR Forum*, 26(2), pp. 26-28, 1992.
- [3] L. L. Scharf, *Statistical Signal Processing: Detection, Estimation and Time Series Analysis*, Addison-Wesley, 1991.
- [4] S. T. Dumais, G. W. Furnas & T. K. Landauer, "Indexing by Latent Semantic Analysis," *Journal of the American Society for Information Science*, 1990, pp. 391-407.

- [5] R. A. Harshman & M. E. Lundy, "Data preprocessing and the extended PARAFAC model," *Research Methods for MultiMode Data Analysis*, edited by H. G. Law et. al., 1984.
- [6] D. Hull, "Improving Text Retrieval for the Routing Problem using Latent Semantic Indexing," *Proc SIGIR '94*, 1994, pp. 282-291.
- [7] H. Hotelling, "Analysis of a Complex Statistical Variables into Principal Components," *Journal of Educational Psychology*, 1933, pp. 417-441, 498-520.
- [8] S. Haykin, *Adaptive Filter Theory*, Prentice Hall, 1986.



Rutgers Interactive Track at TREC-5

N.J. Belkin* , A. Cabezas, C. Cool, K. Kim, K.B. Ng,
S. Park, R. Pressman, S. Rieh, P. Savage, H. Xie

School of Communication, Information & Library Studies
Rutgers University
4 Huntington Street
New Brunswick, NJ 08901-1071
belkin@scils.rutgers.edu

Abstract

The Interactive Track investigation at Rutgers concentrated primarily on three factors: the searchers' uses and understandings of relevance feedback and ranked output, and the utility of relevance feedback for the interactive track task; the searchers' understandings of the interactive track task; and performance differences based on topic characteristics and searcher and order effects. Our official results are for twelve searchers, each of whom did searches on six different topics.

1. Introduction

The Rutgers TREC-5 Interactive Track group had some substantial difficulties in performing the investigation that we originally wanted to do. We had originally intended to compare a system with relevance feedback based on positively-judged documents only, with a system which used both positive and negative judgements in the feedback process. For both systems, following the results of Koenemann (1996), we intended to have user control over which terms were added to the query through the relevance feedback process. For a variety of reasons, we were unable to construct the system which could take account of negative relevance judgements. Our fall-back position was then to test user controlled feedback versus ordinary relevance feedback. For this, we ran into incompatibility snags between the interface structure that we had available for doing this, and the new version of the search system that we wanted to use. Our next, and final fallback position, then, was to use our existing TREC-4 system, in order to investigate:

- the searchers' understanding of the TREC-5 interactive track task;
- the understanding, use and utility of positive-only relevance feedback for this task;
- effects of topic order, difficulty and domain on performance; and,
- the range of performance by different searchers on the same topics.

Because we got to our final fall-back position only just before our experiment had to begin, we did not have time to modify our existing system (INQUERY 2.1p3) to index two of the new databases which were added for the adhoc task this year whose format differed somewhat from previous years (Congressional Record and Federal Register 1994). Therefore, the results that we report here can only be considered partial, since they exclude any access to these two databases.

Our excuses having been made, we can now get to what we *were* able to do, and how we did it.

* To whom all correspondence concerning this paper should be addressed

2. Methods

For a description of the system which we used for TREC-5 (INQUERY 2.1p3, with a local interface, called RU-INQUERY) see Belkin, *et al.* (1996), the report of our TREC-4 study.

We attempted to follow the guidelines for the TREC-5 interactive track quite strictly, to the extent that this was possible. Those guidelines call for each searcher to perform three searches in a control condition, which was to be a version of PRISE, and another three searches in the test condition of the local study. The search topics were organized in four blocks of three searches each, and for each searcher, the order of blocks was specified in a permuted design (*i.e.* B(block) 1 then B2; B2 then B3; B3 then B4; B4 then B1), leading to a minimum of four searchers to complete the design. Unfortunately (sorry, excuses again) we were unable to get PRISE running in time for the beginning of the study, and so the control versus test aspect of the study was no longer relevant. However, since we were interested topic order effects, we maintained this general design, and duplicated the entire sequence three times (*i.e.* using twelve searchers).

We recruited fourteen volunteer subjects to take part in the study, from the community of information professionals in New Jersey, and from the students and faculty of the School of Communication, Information and Library Studies and of the Computer Science Department at Rutgers University. Results from the first two subjects are not reported, however, because they used a slightly different version of the tutorial than the other twelve searchers. Details of the subject characteristics are in the Rutgers Interactive Track "Rich Format" description.

The experimental sessions were held at our lab at Rutgers University, and took about 3 1/4 hours total time. On arrival, the subjects were administered a questionnaire concerned with demographic and educational factors, and previous experience with IR systems. We then applied a structured interview, aimed at identifying their understanding of the TREC-5 interactive track task, and at how they would go about performing this task in a system with which they were familiar. They then did a hands-on structured tutorial in the use of RU-INQUERY, which incorporated an example of the interactive track task. Then, for each search topic, the subjects were handed a sheet of paper with the general instructions for the task, and the specific instructions for that topic, and were told that they had twenty minutes to perform the task for that topic. They were also given, at this time, a separate sheet on which they were asked to list the "aspects" of the topic as they identified them. At the same time, the subjects were instructed to "think aloud" during the search, and this talk was recorded on a videotape of the monitor during the search. In addition, the entire search interaction was logged. Having finished the search, the subjects were asked to complete a brief questionnaire about that search, giving self-reports on familiarity with topic, difficulty of search, satisfaction with search results, confidence in aspect identification, and time enough to do an effective search. This process was repeated for the next two searches in the block, after which the subjects were asked if they wanted to take a break. After the break, the same process was repeated for the three searches of the second block. After all six searches were completed, the subjects were administered an exit questionnaire and interview, whose foci were: the understanding and use of relevance feedback in their searches, and its utility for the interactive track task; and their understanding and experience of the task itself. All of the data-collection instruments are shown in Appendix I.

3. Results

The basic performance result for the interactive task is the so-called *aspect recall*. The task itself was to identify as many aspects of the specific topic as possible, on the basis of the literature, saving at least one document which discussed one or more of the aspects. The basic result that was returned to TREC then, was the list of saved documents, in the order saved, plus the total time for the search (as measured from the time the subjects were given the topics, until they said they were finished). The TREC assessors, bless them, were asked to consider the documents which had

Search	Searcher	Time	Saved	Prec	Recall
254I-003-1	c	1095	12	75	78
254I-005-2	e	1260	13	15	44
254I-007-1	g	1075	3	67	33
254I-011-1	k	1108	9	78	44
254I-012-2	l	1246	7	71	44
254I-014-2	n	1043	7	71	44
254 mean		1137.8	8.5	62.8	47.8
255I-003-2	c	1141	16	38	81
255I-004-1	d	1280	1	100	15
255I-006-2	f	1214	5	20	19
255I-008-1	h	1242	2	100	23
255I-009-2	i	1330	2	100	19
255I-012-1	l	1289	6	50	15
255 mean		1249.3	5.3	68	28.7
256I-004-2	d	1222	5	20	14
256I-005-1	e	1132	3	0	0
256I-007-2	g	1013	10	40	71
256I-009-1	l	1163	2	50	29
256I-010-2	j	617	11	9	14
256I-013-1	m	1092	2	50	43
256 mean		1039.8	5.5	28.2	28.5
258I-003-2	c	1182	28	64	79
258I-004-1		1138	7	29	38
258I-006-2	f	1225	7	43	38
258I-008-1	h	1094	17	53	67
258I-009-2	i	1231	5	80	54
258I-012-1	l	1214	10	90	67
258 mean		1180.7	12.3	59.8	57.2
260I-006-1	f	1333	1	100	50
260I-008-2	h	1211	8	62	83
260I-010-1	j	1195	12	17	50
260I-011-2	k	1178	2	50	50
260I-013-2	m	751	3	33	33
260I-014-1	n	1304	0	0	0
260 mean		1162	4.3	43.7	44.3
264I-006-1	f	1217	3	67	6
264I-008-2	h	1182	11	100	29
264I-010-1	j	1344	15	87	47
264I-011-2	k	1127	13	100	47
264I-013-2	m	830	3	100	18
264I-014-1	n	1298	15	100	71
264 mean		1166.3	10	92.3	36.3

Search	Searcher	Time	Saved	Prec	Recall
274I-004-2	d	1047	8	100	64
274I-005-1	e	1385	6	83	73
274I-007-2	g	712	14	100	73
274I-009-1	i	1247	5	100	73
274I-010-2	j	1243	36	94	100
274I-013-1	m	1191	6	100	64
274 mean		1137.5	12.5	96.2	74.5
284I-003-1	c	1188	22	50	52
284I-005-2	e	1157	13	69	44
284I-007-1	g	1015	9	44	24
284I-011-1	k	1018	17	41	44
284I-012-2	l	1201	18	67	56
284I-014-2	n	818	10	80	36
284 mean		1066.2	14.8	58.5	42.7
286I-006-1	f	1226	4	50	44
286I-008-2	h	1121	14	7	44
286I-010-1	j	1013	9	44	56
286I-011-2	k	1251	11	82	56
286I-013-2	m	590	5	60	44
286I-014-1	n	1087	15	73	56
286 mean		1048	9.7	52.7	50
292I-003-1	c	1067	19	21	56
292I-005-2	e	1209	22	36	41
292I-007-1	g	1254	16	31	47
292I-011-1	k	1213	19	47	47
292I-012-2	l	1212	22	32	44
292I-014-2	n	945	11	45	38
292 mean		1150	18.2	35.3	45.5
293I-004-2	d	1357	0	0	0
293I-005-1	e	1261	11	45	100
293I-007-2	g	1337	1	100	17
293I-009-1	i	1136	5	20	17
293I-010-2	j	964	6	0	0
293I-013-1	m	1088	2	50	17
293 mean		1190.5	4.2	35.8	25.2
299I-003-2	c	1195	18	39	73
299I-004-1	d	1251	3	33	47
299I-006-2	f	1337	6	7	7
299I-008-1	h	1140	9	44	87
299I-009-2	i	1236	2	50	40
299I-012-1	l	1157	7	57	60
299 mean		1219.3	7.5	38.3	52.3
Overall		1180.3	9.4	56	43.8

Table 1. Performance data by topic and searcher.

already been judged relevant to the topic in the adhoc task relevance assessment, as well as those retrieved by the interactive track participants, and to identify all of the aspects of the topic, and the documents in which each aspect was discussed. Then, aspect recall was computed by comparing the list of saved documents for each search with the list of document-aspect tuples identified by the assessors. If documents were saved which, *in toto*, discussed all of the aspects, aspect recall was 100. No penalty, nor advantage, was assigned for multiple identification of the same aspect. Precision was computed in the normal manner: as the percentage of saved documents which treated at least one aspect, *i.e.* were relevant.

Table 1 shows the results for all six searches for each topic, with mean values for each topic and for the study overall.

Other analyses of the data will be presented at the meeting, as will some thoughts that they will have engendered about the nature of the interactive track task, the evaluation methodologies, and what one can hope to learn from this type of evaluation study.

4. References

Belkin, N.J., Cool, C., Koenemann, J., Ng, K.B., Park, S. (1996) Using relevance feedback and ranking in interactive searching. In: D. Harman, ed. *TREC-4. Proceedings of the Fourth Text Retrieval Conference*. Washington, D.C., GPO: in press.

Koenemann, J. (1996) *Relevance feedback: Usage, usability, utility*. Ph.D. Dissertation, Graduate Program in Psychology, Rutgers University, New Brunswick, NJ

Appendix I. Experimental Materials

I.1 Entry questionnaire

RUTGERS TREC-5 INTERACTIVE SEARCHING STUDY SEARCHER QUESTIONNAIRE

Please list all the college/university degrees that you have (or expect to have):

_____	_____	_____
Degree	Major	Date

_____	_____	_____
Degree	Major	Date

_____	_____	_____
Degree	Major	Date

_____	_____	_____
Degree	Major	Date

What is your age?

- | | | |
|-----------------------------------|--------------------------------|----------------------------------|
| <input type="checkbox"/> Under 21 | <input type="checkbox"/> 31-40 | <input type="checkbox"/> 51-60 |
| <input type="checkbox"/> 21-30 | <input type="checkbox"/> 41-50 | <input type="checkbox"/> Over 60 |

What is your gender?

- ☐ Female
☐ Male

Please circle the appropriate number...

How much experience have you had...	None	1	2	3	4	5	Some	6	7	8	A great deal
searching on computerized library catalogs	1	2	3	4	5						
searching on CD ROM systems, e.g., Infotrac, Grolier	1	2	3	4	5						
searching on commercial online systems, e.g., Dialog, Lexis, BRS Afterdark	1	2	3	4	5						
searching on world wide web browsers, e.g., Mosaic, Netscape	1	2	3	4	5						
searching on other systems, please specify the system: _____	1	2	3	4	5						
searching full-text databases	1	2	3	4	5						
searching in ranked-output information retrieval systems	1	2	3	4	5						
searching in information retrieval systems that provide automatic relevance feedback	1	2	3	4	5						
using a mouse-based interface	1	2	3	4	5						

Overall, for how many years have you been doing online searching? _____ years

Who have you performed searches for?

- ☐ Yourself only
- ☐ Others only (e.g., as an intermediary)
- ☐ Yourself and others

Have you participated in previous TREC Searching Studies?

- ☐ Yes
- ☐ No

I.2. Pre-search interview

RUTGERS TREC-5 INTERACTIVE SEARCHING STUDY PRE-SEARCH INTERVIEW

In order to understand the different searching experiences of our participants, we would like you to answer a couple of questions about the methods that you typically use when you do online searching. When you answer these questions, please try to give us as much detail as you can. Please use this worksheet for any notes that you wish to make. (*Hand worksheet to searcher.*)

Imagine that you are interested in learning about the different alternative sources of energy for automobiles. You decide to investigate the literature by using a computerized database of newspaper articles that is available for your use. Since you are interested in identifying as many "aspects" of this topic as possible, you will want to identify each one of the different alternative sources of energy for automobiles, including gasoline additives that decrease pollution or reduce oil consumption.

1. What steps would you take in order to perform a search that would identify as many "aspects" as possible for this topic? Describe your overall approach by listing what you would do first, and then describe each of the steps that you would follow after that.
2. How would you decide that your search is finished?

I.3. Search evaluation form

RUTGERS TREC-5 INTERACTIVE SEARCHING STUDY SEARCH EVALUATION FORM

TOPIC NUMBER _____

Please answer the following questions, as they relate to this specific topic.

Please circle the appropriate number...

To what extent...	Not at all		Marginally		Extremely
are you familiar with this topic?	1	2	3	4	5
was it difficult to do this search?	1	2	3	4	5
are you satisfied with your search results?	1	2	3	4	5
are you confident that you identified all the possible aspects for this topic?	1	2	3	4	5
did you have enough time to do an effective search?	1	2	3	4	5

I.4. Topic description and task instructions

RUTGERS TREC-5 INTERACTIVE SEARCHING STUDY TOPIC DESCRIPTION

Topic 256i

Negative reactions to reduced requirements for college undergraduate core studies

GENERAL INSTRUCTIONS

Now we would like you to identify as many aspects as possible for each topic that will be presented to you. You will be given 20 minutes to search for each topic's aspects. Please save one document for each of the aspects that you identify. If you save one document that contains many aspects, try not to save additional documents that contain only those aspects, unless a document contains additional aspects as well.

Carefully read each description and narrative for each topic because the interpretation of "aspects" changes from topic to topic. For example, aspects can refer to different developments in a field, to different instances in which an event can occur, or to different kinds of treatments -- as it did in the high blood pressure example.

SPECIFIC INSTRUCTIONS

Topic

Colleges for a long time have been reducing their requirements in such core subjects as history, literature, philosophy, and science. Criticism of this trend has occurred.

Aspects

Please save at least one document that identifies EACH DIFFERENT criticism of this trend. If one document discusses several criticisms, then you need not save other documents that repeat those aspects, since your goal is to identify the different criticisms that have been made.

Narrative

To be relevant, a document will provide negative opinions/facts concerning the fact that colleges have reduced their basic requirements for the granting of degrees to undergraduates.

I.5. Aspect identification worksheet

RUTGERS TREC-5 INTERACTIVE SEARCHING STUDY SEARCHER WORKSHEET

Searcher # _____

Please use this sheet of paper to write down any notes that you'd like to make during your search and to list the aspects as you identify them for this topic.

Topic 254i:

I.6. Post-search interview

Searcher # _____

RUTGERS TREC-5 INTERACTIVE SEARCHING STUDY POST-SEARCH INTERVIEW

So we can have a better understanding of your overall searching experience, I'd like to ask you some final questions about your experiences today. In order to answer the first set of questions, I'd like you to use this scale (*hand scale sheet to participant*). In this scale a "1" means "not at all", a "3" means "marginally", and a "5" means "to a great extent".

1. To what extent did you understand how to use Relevance Feedback?

Not at all		Marginally		To a great extent	
1	2	3	4	5	

Why is that?

2. To what extent did you use Relevance Feedback during your searches?

Not at all		Marginally		To a great extent	
1	2	3	4	5	

rating = 1, 2 or 3: Why didn't you use it more?

rating = 4 or 5: Why did you use it so much?

3. To what extent did you find Relevance Feedback useful during your searches?

Not at all		Marginally		To a great extent	
1	2	3	4	5	

rating = 1, 2 or 3: Why wasn't it useful? What would have made it more useful?

rating = 4 or 5: Why did you find it useful?

4. To what extent did Relevance Feedback improve your ability to identify different aspects of the topics?

Not at all		Marginally		To a great extent	
1	2	3	4	5	

rating = 1, 2 or 3: What would have made it more useful to identify different topic aspects?

rating = 4 or 5: Why did you find it useful in improving your ability to identify different topic aspects?

5. To what extent was it helpful to have Ranked Output in your searches?

Not at all		Marginally		To a great extent	
1	2	3	4	5	

rating = 1, 2 or 3: What would have made Ranked Output more helpful?

rating = 4 or 5: Why did you find Ranked so helpful?

6. To what extent did you find this task different from other searching tasks that you typically perform?

Not at all		Marginally		To a great extent	
1	2	3	4	5	

rating = 1, 2 or 3: In what ways was this task different from other searching tasks that you typically perform?

rating = 4 or 5: In what ways was this task similiar to other searching tasks that you typically perform?

7. To what extent were you able to understand the nature of the task?

Not at all		Marginally		To a great extent	
1	2	3	4	5	

rating = 1, 2 or 3: What did you find confusing?

rating = 4 or 5: Why did you find this easy to understand?

8. Do you have any other comments about your experiences with RU-INQUERY?



Interactive Substring Retrieval (MultiText Experiments for TREC-5)

Charles L. A. Clarke*

Gordon V. Cormack

MultiText Project
Department of Computer Science
University of Waterloo
Waterloo, Ontario N2L 3G1, Canada
mt@plg.uwaterloo.ca
<http://multitext.uwaterloo.ca>

Abstract

Queries for TREC-5 were formulated in the GCL query language using an interactive system that showed short passages containing relevant terms. Solutions to the queries were ranked by the shortest substring method introduced at TREC-4, resulting in good precision/recall performance in the adhoc and routing tasks. Performance results were found to be insensitive to a document length normalization adjustment. Shortest substring ranking was augmented by the use of a progression of successively weaker queries to improve recall, but this augmentation provided only a slight improvement to overall retrieval effectiveness.

1 Introduction

For TREC-5, the MultiText Project focused on the development of interaction methods for query creation and refinement using a ranking technique introduced at TREC-4. In the MultiText system, ranking is based entirely on term proximity, with queries expressed in a boolean subset of the general structured text retrieval language GCL [3, 4, 5]. In the GCL query language, solutions to queries represent intervals of the underlying text and are not restricted to reporting pre-defined elements such as documents, paragraphs or sentences. For ranking purposes, a document's score is based on the size and density of the query solutions contained within it. The development and initial validation of this ranking technique was the subject of our participation in TREC-4 [6].

Since relevance depends on term proximity, the technique may be used to identify small, relevant passages. During interactive query development, these passages are presented to the user for the selection of additional query terms and to drive other refinements. Initial exploration of this interaction method was the primary subject of our TREC-5 work.

Hearst [10] argues that efforts in interactive information retrieval should be directed toward "providing the user with information that is informative and compact enough that it can be interpreted swiftly." Toward this end, Hearst presents a visualization tool that summarizes term distribution and density within documents in an iconic form. Interactive retrieval and presentation of short relevant passages may be considered as a complementary technique, providing the user with specific textual context. Along these lines, interactive passage retrieval has been used to speed up

* Author's current address: Department of Electrical and Computer Engineering, University of Toronto.

the process of making relevance judgements [12] and has been suggested for avoiding user overload when dealing with long documents [14].

Apart from this use in interactive retrieval, passage-level evidence has been widely used to improve retrieval effectiveness [1, 2, 11, 14, 17, 18]. Research in this area has generally assumed that documents must be divided *a priori* into passages. Wilkinson [17] based passage retrieval on explicit structural elements — such as abstracts, titles and summaries — described by the internal document markup. Callan [2] compared passages based on paragraphs with passages based on fixed-length text segments. Wilkinson and Zobel [18] based retrieval on a variety of passage types, including sentences, paragraphs, sequences of paragraphs, and sections defined by document markup. Hearst and Plaunt [11] used a similarity measure between successive blocks of text to partition documents according to their subtopic structure.

In contrast, our approach to interactive retrieval bases the decomposition into passages on the query itself. High-scoring passages satisfying the query are presented to the user, who may then select additional terms from these passages for inclusion in the query. This approach to interactive query expansion, with the user selecting from suggestions made by the system, is similar to that taken by Koenemann and Belkin [13], using terms suggested by relevance feedback, and by Harman [7], using terms gathered from relevance feedback, reverse stemming and nearest neighbors procedures.

2 The GCL Query Language

The GCL query language was initially developed for structural document retrieval. Text in a database is viewed as a long sequence of words. Solutions to queries are the intervals of this text sequence that satisfy the conditions of the query, expressed as start and end positions within the sequence. In order to limit the total number of solutions, we restrict the solution set to include only those intervals that do not contain smaller intervals that satisfy the query conditions. This *shortest substring* rule provides linearity and ordering properties that make efficient query evaluation possible and is central to the document scoring technique. Structural information — for example, the start and end positions of documents, titles, paragraphs, headlines and document identifiers — are indexed at appropriate positions within the word sequence but do not necessarily occupy positions in the sequence.

The simplest GCL query is a phrase, for example:

```
"heart attack"
```

The solution to this query is the set of intervals corresponding to the locations of the phrase within the word sequence.

An ordering operator ("...") is provided to link the start and end positions of text intervals defined by sub-queries. The query

```
"<title>"...</title>"
```

links the start and end tags for titles, and has as its result the set of all titles. Document boundaries require no special treatment. The query

```
"<doc>"...</doc>"
```

has as its result the set of all documents. The shortest substring rule guarantees that the solution set contains only single documents. Start and end tags that are more than a single document apart

are not linked. The language includes four *containment operators*— `containing`, `not containing`, `contained in`, and `not contained in`— which may be used to query structural relationships. The query

`("angiogram"... "bypass surgery") contained in ("<doc>..."</doc>")`

finds occurrences of “angiogram” followed by “bypass surgery” within the same document. Note that elements in the solution set for this query are text intervals, not documents.

The query language includes the boolean operators “and” and “or” interpreted over intervals of text under the shortest substring rule. Rather than selecting members from a document set, as might normally be expected, the query

`"angiogram" and ("bypass surgery" or "angioplasty")`

specifies intervals of text that satisfy the boolean expression. Ranking of documents containing solutions to this query would be based on the length and density of these solution intervals.

Further details of the query language, its implementation and the underlying theory may be found elsewhere [3, 4, 5].

3 Shortest Substring Ranking

The ranking technique is based on two assumptions:

- A) the shorter a solution interval, the greater the likelihood that the interval is relevant
- B) the more solution intervals within a document, the greater the likelihood that the document is relevant

An interval in the text is represented by an range or *extent* (p, q) , where p is the start position of the interval within the word sequence and q is the end position of the interval. An extent is assigned a score using the formula:

$$I(p, q) = \begin{cases} \frac{\mathcal{K}}{|q-p+1|} & \text{if } |q-p+1| \geq \mathcal{K} \\ 1 & \text{otherwise.} \end{cases}$$

A value of 16 for the parameter \mathcal{K} has been found to give reasonable performance. Using this formula, intervals of 16 words or smaller are assigned a score of 1, and larger intervals are scored in proportion to the inverse of their length.

If a document D contains solution extents (p_1, q_1) , (p_2, q_2) , ..., (p_n, q_n) the score for the document is taken to be the sum of the scores of these extents:

$$S(D) = \sum_{i=1}^n I(p_i, q_i)$$

Query evaluation and ranking proceeds in five steps:

1. Determine extents which satisfy the query: Q
2. For each solution extent, either determine the document extent that contains it, or if it overlaps document boundaries eliminate it:

`("<doc>..."</doc>") containing Q`

3. For each document containing solution extents, determine the identifier for the document:

("<docno>"... "</docno>")
contained in ("<doc>"... "</doc>") containing Q)

4. For each document containing solution extents, calculate a document score using the scoring method above.

5. Sort the document identifiers by document score.

Shortest substring ranking has several important properties. A document's score is independent of the characteristics of other documents in the collection, making special consideration for distributed collections unnecessary. Phrase queries, structural queries and boolean queries are naturally integrated in the query model. Any set of document elements that can be defined by a GCL query may be ranked, without any need for supplementary indexing. Since the scoring technique is based on term proximity, short relevant passages may easily be identified. This last property forms the bases for the interaction method.

Apart from the work of the MultiText Project, document ranking based solely on term proximity has been the subject of research at the Australian National University [8, 9]. Although details of the scoring technique differ, that work supports our view that ranking can reasonably be based on term proximity alone.

4 Interactive Substring Retrieval

Under shortest substring ranking, scores are the output of an evaluation procedure executed over predetermined text intervals (e.g. documents). For interactive retrieval, a *score threshold* becomes an input parameter to an evaluation procedure whose result is the smallest intervals achieving the threshold score. Given a query Q , let

$$(p_1, q_1), (p_2, q_2), \dots, (p_m, q_m)$$

be the solution extents for Q ordered by increasing position in the word sequence. Given a threshold score (*Threshold*) the algorithm in Figure 1 computes the shortest intervals that meet or exceed the threshold and lie within a single document.

The algorithm makes a single scan over the solution extents. The variables i and j represent the start and end positions of a range of solution extents. The first **while** loop advances j until the score of the interval between (p_i, q_i) and (p_j, q_j) exceeds the threshold value. The second **while** loop then advances i until any further advance would drop the score below the threshold. All intervals containing (p_i, q_j) meet or exceed the threshold; all intervals contained in (p_i, q_j) have scores lower than the threshold. If (p_i, q_j) is contained within a single document, it is presented to the user.

The solution extents $(p_1, q_1), \dots, (p_n, q_n)$ are generated in word sequence order by GCL and may be generated one at a time as needed for the the threshold passage retrieval algorithm. In turn, the algorithm can generate intervals exceeding the threshold without requiring the full set of solution extents. For interactive query refinement, the algorithm may be used to quickly generate a small number of high-scoring passages for presentation to the user, without a need to invoke the full ranking process.


```

i ← 1;
j ← 0;
Score ← 0.0;
loop
  while Score < Threshold do
    if j = m then stop; end if;
    j ← j + 1;
    Score ← Score + I(pj, qj);
  end while;
  while Score − I(pi, qi) ≥ Threshold do
    Score ← Score − I(pi, qi);
    i ← i + 1;
  end while;
  if {(pi, qj)} contained in "<doc>"..."</doc>" then
    Display (pi, qj);
  end if;
  if i = m then stop; end if;
  Score ← Score − I(pi, qi);
  i ← i + 1;
end loop;

```

Figure 1: Threshold Passage Retrieval

5 MultiText Experiments for TREC-5

The MultiText Project participated in both the routing task and the adhoc task. Queries were developed manually by the authors working in conjunction. Approximately 45 minutes were spent developing a query for each topic. For the routing task the queries were developed interactively over a collection consisting of the documents from TIPSTER disks #2 and #3. Relevance judgements were indexed as part of the document structure and could be directly referenced in queries. For example, the query

```
((("<doc>"..."</doc>")) containing "<+78>") not containing "greenpeace"
```

specifies documents that have been judged relevant to topic 78 that do not contain the word "greenpeace". The query

```
((("<doc>"..."</doc>")) containing "<-78>") containing "greenpeace"
```

specifies documents that have been judged not relevant to topic 78 that contain the word "greenpeace". For the adhoc task the queries were developed interactively over the collection consisting of TIPSTER disk #2 and TREC disk #4. For adhoc query development, interaction was used primarily for term expansion purposes, and the resulting queries are not intended to incorporate specific relevance judgements made during interaction. The final queries would be suitable for use in a future routing task. Besides the authors' personal knowledge of the topics, the only external resources used were Webster's on-line dictionary, the Unix `spell` program, and an on-line list of country, state and city names and state postal abbreviations.

Figure 2 shows our user interface for interactive query development, with the results of a query for topic 272 displayed. The first result is displayed with a small window of additional text surrounding the high-scoring passage and with terms from the query highlighted.

The final query developed for each topic was a compound query consisting of an ordered list of one or more sub-queries (1.79 sub-queries on average). Ranking for each sub-query was first determined separately, and the results were then combined into a final solution set according to the ordering of the sub-query list. The sub-query list was processed in order. If a document was given a non-zero score by a sub-query, it was eliminated from the results of subsequent sub-queries in the final ranking.

This approach reflects a trade-off between a desire for precision and the need to produce 1000 ranked documents. The query appearing at the beginning of the list is intended to be a precise expression of the requirements underlying the topic. Queries occurring later in the list are "weaker" and are intended to pick up a large number of possibly relevant documents. The effects of using these weaker, secondary sub-queries will be examined in the next section.

The number of terms used in the final queries varied from one to more than a hundred. For both tasks, queries contained an average of 44 terms. The large number of terms was partially due to the explicit listing of morphological variants and partially due to the repeated use of a pre-defined sub-query listing 150 geographical place names within the United States.

The official MultiText routing run (`uwgcr0`) and one of the official MultiText adhoc runs (`uwgcx0`) used the scoring method described in Section 3. The second official adhoc run (`uwgcx1`) used a variation of this scoring technique, described in Section 5.2 below, that attempts to normalize scores to account for document length.

The experiments were run on a dedicated DEC Alpha 2000/300 workstation running OSF V3.2 with 64MB of RAM.

MultiText TRECsearch Results

Query name: topic 272

("outpatient" or "outpatients" or "ambulatory") and ("surgery" or "surgical" or "surgeon" or "surgeons")

AP880324-0007

[...] Hospital Stay, Researcher
 Reports</HEAD>
 <NOTE>Eds: Also in Thursday AMs report.</NOTE>
 <BYLINE>By MALCOLM RITTER</BYLINE>
 <BYLINE>AP Science Writer</BYLINE>
 <DATELINE>NEW YORK (AP) </DATELINE>
 <TEXT>
 An experimental treatment for enlarged prostate glands shows promise as an outpatient alternative to surgery that hospitalizes about 400,000 American men a year, researchers say.
 An enlarged prostate makes urination difficult by constricting the urinary tract, and the experimental procedure widens the narrowed portion by inflating a balloon, said Flavio Castaneda of the University [...]

AP880412-0054

[...] <DATELINE>NEW YORK (AP) </DATELINE>
 <TEXT>
 James E. Olson, chairman and chief executive of American Telephone & Telegraph Co., has relinquished his duties

Figure 2: Interactive Retrieval Interface

5.1 Sub-Query Lists

Each query used in the MultiText runs consisted of an ordered list of one or more sub-queries. The first of these sub-queries was intended to be a “high-precision” query, an accurate expression of the user requirements embodied in the topic. The additional sub-queries were intended to increase overall recall. After the results of the first query were ranked, the remaining sub-queries were evaluated in order. New documents introduced by these sub-queries were ranked after the documents matching the first sub-query.

As part of our TREC-5 work, we performed three unofficial runs intended to assess the impact of these additional sub-queries. Recall-precision curves for the runs are plotted in figure 3, along with the curve for the corresponding official TREC-5 adhoc run (uwgcx0).

For the first unofficial run (plotted with “+”) we eliminated the additional sub-queries, and used only the first sub-query for retrieval and ranking. As expected, overall recall was reduced. Our official run retrieved 3072 relevant documents, but the first sub-queries alone retrieved only 2781 relevant documents. The second unofficial run (plotted with “□”) used all sub-queries but did not use shortest substring ranking. Documents were ranked according to the sub-query that first retrieved them, with documents first retrieved by earlier sub-queries given higher scores. All documents first retrieved by a particular sub-query were given the same score, and their relative ordering was determined only by their position in the database. The difference in retrieval effectiveness between this run and the official run is dramatic, showing substantially lower precision at all recall levels.

The third unofficial run (plotted with “×”) used the first sub-query only, treating it as a traditional boolean query. For this run, documents were ordered only by their position in the database. The improvement seen in retrieval effectiveness between this run and the first unofficial run (“+”) is entirely due to shortest substring ranking.

5.2 Document Length Normalization

Users of ranked retrieval in the MultiText system have noticed a slight bias for longer documents. Solely because of length, a long document may contain many more high-scoring solution extents than a short document with similar relevance, and receive an inappropriately higher score. It has been noted by others [15] that in the TREC collection a longer document is more likely to be relevant than a shorter document. Nonetheless, we viewed the lack of *any* document length normalization as a possible deficiency in the scoring method.

In preliminary experiments, normalization of scores by dividing by document length appeared to excessively penalize long documents with short relevant passages. However, a weak normalization, dividing by the square root of the document length, appeared to improve precision among high ranking documents.

The high variance in document length found in the adhoc collection provided an opportunity to test this document normalization technique. For our second official adhoc run (uwgcx1) documents were scored using the formula

$$S'(D) = \frac{S(D)}{\sqrt{|D|}}$$

where $|D|$ is the length of document D . The other official run (uwgcx0) used the scoring procedure of Section 3.

Document length normalization had little impact on overall retrieval effectiveness. Figure 4 plots recall and precision for the two runs. The largest difference in precision occurs at the 10% recall level, where the normalized run achieves a slightly higher precision (0.6024 vs. 0.5833).

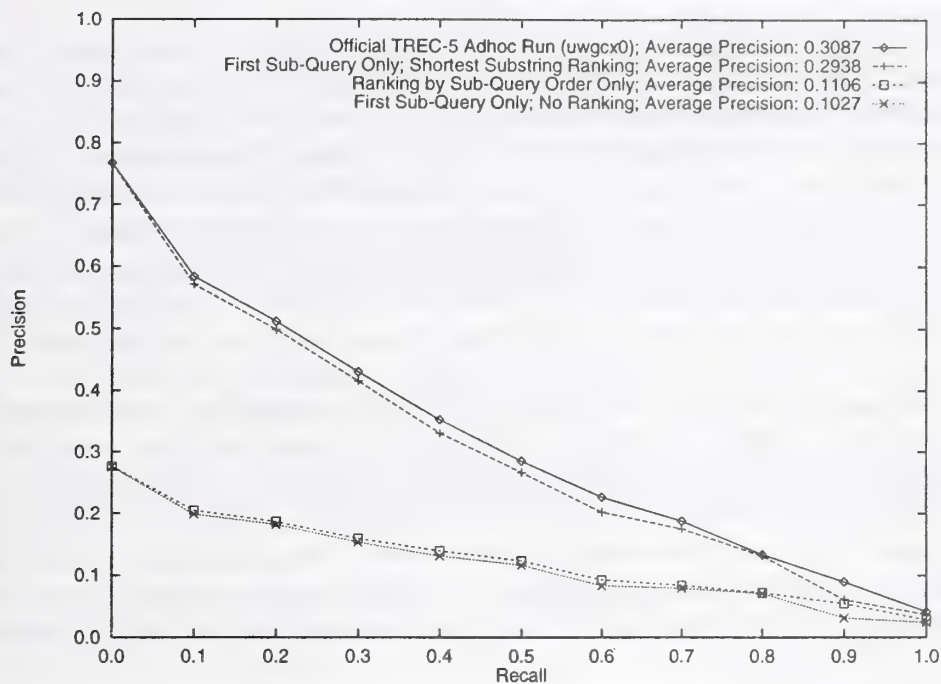


Figure 3: Effects of Sub-Query Lists

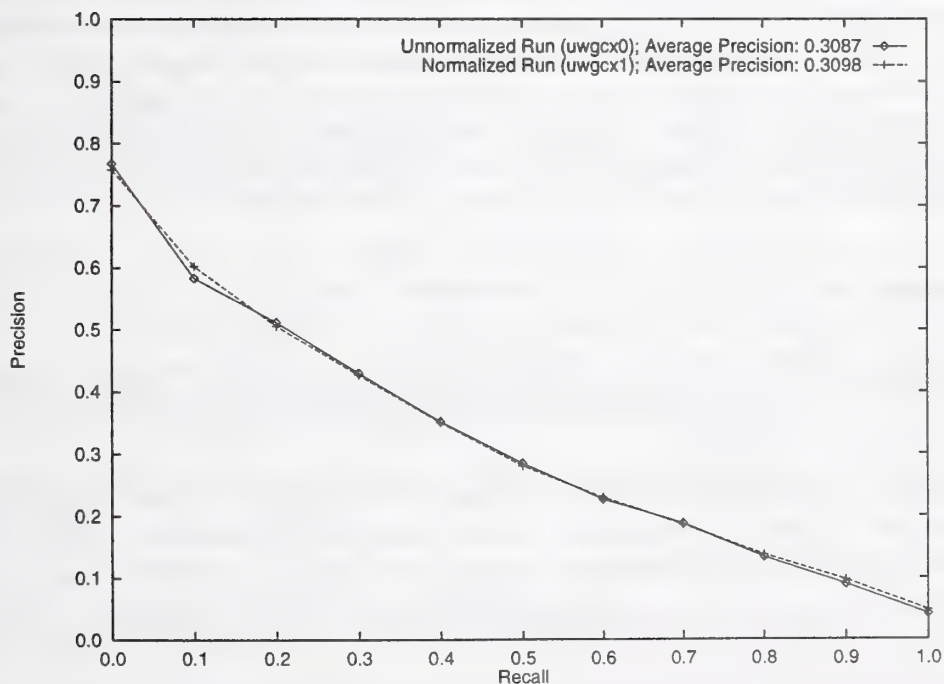


Figure 4: Effects of Document Length Normalization

6 Concluding Remarks

The work described in this paper was undertaken as part of the MultiText Project. The MultiText Project develops and prototypes new technologies associated with large-scale document database systems. Research issues include fault tolerance, distributed data management, dynamic update, and support for heterogeneous document structure. Prototype development is based around a federated architecture in which a group of small specialized servers under common administration present the unified services of a large, highly-reliable document database system to external clients. Emphasis is placed on flexibility. A large-scale system must store documents in wide variety of formats with multiple user interfaces. In addition, the federated architecture can provide an increase in efficiency and reliability if appropriate data organization and replication methods are used.

Part of the ongoing work of the MultiText Project is directed toward the creation of tools for interactive query refinement. We are particularly interested in query expansion techniques, including the work described in this paper and the use of stemming algorithms for interactive term expansion [16]. Our TREC-5 work provided preliminary experience with our approach. For TREC-6 we hope to perform a more substantial evaluation of our tools, particularly through participation in the Interactive and High-Precision tracks. In addition, given our overall focus on large-scale document database systems, we are planning to participate in the Very Large Collection track.

Acknowledgements

Rob Good provides technical and system administration support for the MultiText Project and gave some much-needed last-minute assistance with data indexing for our TREC-5 work. Other people who have contributed to the MultiText Project include Forbes Burkowski, Elizabeth Tudhope, Nikita Borisov and Chris Palmer. The first author would like to thank Rob Pitt of Canning & Pitt Associates, St. John's, Newfoundland for access to office space and computer facilities during an unexpected and lengthy stay in that city overlapping the TREC-5 data submission deadline.

The MultiText Project receives its primary funding from the Information Technology Research Centre, Ontario. Additional funding was provided by the Natural Sciences and Engineering Research Council of Canada.

References

- [1] James Allan. Relevance feedback with too much data. In *18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'95)*, pages 337–343, Seattle, July 1995.
- [2] James Callan. Passage-level evidence in document retrieval. In *17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'94)*, pages 302–310, Dublin, July 1994.
- [3] Charles L. A. Clarke. *An Algebra for Structured Text Search*. PhD thesis, University of Waterloo, Waterloo, Ontario, 1996.
- [4] Charles L. A. Clarke, G. V. Cormack, and F. J. Burkowski. An algebra for structured text search and a framework for its implementation. *The Computer Journal*, 38(1):43–56, 1995.

- [5] Charles L. A. Clarke, Gordon V. Cormack, and Forbes J. Burkowski. Schema-independent retrieval from heterogeneous structured text. In *Fourth Annual Symposium on Document Analysis and Information Retrieval (SDAIR'95)*, pages 279–289, Las Vegas, April 1995.
- [6] Charles L. A. Clarke, Gordon V. Cormack, and Forbes J. Burkowski. Shortest substring ranking (MultiText experiments for TREC-4). In *Fourth Text REtrieval Conference (TREC-4)*, Gaithersburg, Maryland, November 1995.
- [7] Donna Harman. Towards interactive query expansion. In *11th International Conference on Research and Development in Information Retrieval (SIGIR'88)*, pages 321–331, Grenoble, France, June 1988.
- [8] David Hawking and Paul Thistlewaite. Proximity operators — So near and yet so far. In *Fourth Text REtrieval Conference (TREC-4)*, Gaithersburg, Maryland, November 1995.
- [9] David Hawking and Paul Thistlewaite. Relevance weighting using distance between term occurrences. Computer Science Technical Report TR-CS-96-08, Australian National University, August 1996.
- [10] Marti A. Hearst. TileBars: Visualization of term distribution information in full text information access. In *ACM SIGCHI Conference on Human Factors in Computing Systems (CHI'95)*, pages 59–66, Denver, May 1995.
- [11] Marti A. Hearst and Christian Plaunt. Subtopic structuring for full-length document access. In *16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'93)*, pages 59–68, Pittsburgh, July 1993.
- [12] Daniel Knaus, Elke Mittendorf, Peter Schäuble, and Pádraic Sheridan. Highlighting relevant passages for users of the interactive SPIDER retrieval system. In *Fourth Text REtrieval Conference (TREC-4)*, Gaithersburg, Maryland, November 1995.
- [13] Jürgen Koenemann and Nicholas J. Belkin. A case for interaction: A study of interactive information retrieval behavior and effectiveness. In *Proceeding of the ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 205–212, Vancouver, BC, April 1996.
- [14] Gerard Salton, J. Allan, and C. Buckley. Approaches to passage retrieval in full text information systems. In *16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'93)*, pages 49–58, Pittsburgh, July 1993.
- [15] Amit Singhal, Gerard Salton, Mandar Mitra, and Chris Buckley. Document length normalization. *Information Processing and Management*, 32(5):619–633, 1996.
- [16] Elizabeth Tudhope. Query based stemming. Master's thesis, University of Waterloo, Waterloo, Ontario, 1996.
- [17] Ross Wilkinson. Effective retrieval of structured documents. In *17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'94)*, pages 311–317, Dublin, July 1994.
- [18] Ross Wilkinson and Justin Zobel. Comparison of fragmentation schemes for document retrieval. In *Third Text REtrieval Conference (TREC-3)*, pages 81–84, Gaithersburg, Maryland, November 1994.



V-Twin: A Lightweight Engine for Interactive Use

Daniel E. Rose

Curt Stevens

Apple Research Laboratories
Apple Computer, Inc.
One Infinite Loop, MS 301-4A
Cupertino, CA 95014

{rose|curt}@apple.com

Abstract

This paper describes V-Twin, an information access toolkit designed to provide indexing and search capabilities for a variety of applications. We discuss the phenomenon of very short queries generated by users of interactive search services, and summarize a new technique we are using in V-Twin to handle these queries more effectively. We then present some results based on V-Twin's performance at the TREC-5 ad hoc task. V-Twin achieved a high level of performance despite having much lower index overhead and memory footprint than other systems participating in TREC.

1. V-Twin Background

V-Twin is the code name of the Apple Information Access Toolkit (AIAT), originally created in Apple Research Labs and now available for licensing by Apple developers. The toolkit consists of a library of C++ classes designed to be linked to a front-end application. V-Twin is based on the standard vector model of information retrieval, with a variant of *tf x idf* weighting and length normalization. V-Twin was developed not only to facilitate our own research on experimental information access systems but also to be used in Apple's products¹ and those of our developers. In order to meet the needs of this diverse community, it was designed to be scalable with respect to speed, disk usage, and in particular, memory usage.

One of the primary applications we envisioned for V-Twin was full-text relevance-ranked searching as part of the Macintosh Finder (the file system interface). This meant that it had to work effectively on all the configurations of Macintosh offered by Apple. At the time of development this included laptops and portables based on both RISC-based PowerPC and the older Motorola 68K families of microprocessors, and as little as 8 MB of memory.

¹ More information about V-Twin product plans is available on the world-wide web at <http://www.research.apple.com/research/tech/V-Twin>.

Index overhead had to be kept low (between 15 and 30% of the source text, depending on which features of the system were used), and the indexes had to support in-place updating. Since low-end users of Macintosh computers typically do not use virtual memory, V-Twin-based applications are able to constrain the amount of memory used for indexing; a larger memory allocation produces better indexing performance.

2. Optimizing for Short Queries

V-Twin was intended at the outset to be used in interactive, real-time environments by ordinary users with little or no previous experience with searching. This meant not only designing the APIs to provide easy access to features we wanted to encourage (such as relevance feedback), but also designing interfaces that facilitated their use. One of the applications we built, an indexing and search CGI for MacOS-based web servers called "Apple e.g.", was freely distributed on the Internet² and tested on several of Apple's web sites. Like many V-Twin-based applications, Apple e.g. shows a graphical score bar for each hit as well as a list of query terms found in the document, ordered by their importance.

In looking at the query logs and listening to feedback from users, we were struck by two things. (Note: A detailed discussion of these issues may be found in another paper [Rose & Cutting, 1996]). First, out of over 10,000 queries issued by about 4500 distinct users, the majority of queries (53%) consisted of only one word, and nearly all the queries were three words or less. We later learned that these results are consistent with (if somewhat more extreme than) those found by other search services, as shown in Table 1. In every case, at least 87% of the queries were three words or less.

Second, when a document containing one query term scored higher in relevance than a document containing two query terms, users considered it an error. One reaction to this might be to conclude that users don't have experience with ranked searching, and try to educate them to set their expectations. However, since the goal of a retrieval system is to estimate how much each item is relevant *to the user*, we decided to adjust the system's scoring to be more consistent with users' expectations. Specifically, we wanted to reorder the hits without adversely affecting system performance.

Note that we did not conclude that users really wanted a Boolean query or assumed that a Boolean AND operator was implied. They were content to have some hits with all the query terms and some hits with only a subset. It was the ordering of certain hits they objected to. It was clear that for very short queries, the coordination level (the number of matching query terms) of a document was of primary importance. Nevertheless, for longer queries, and for relevance feedback queries, the coordination level was increasingly

² http://cybertech.apple.com/apple_eg.html

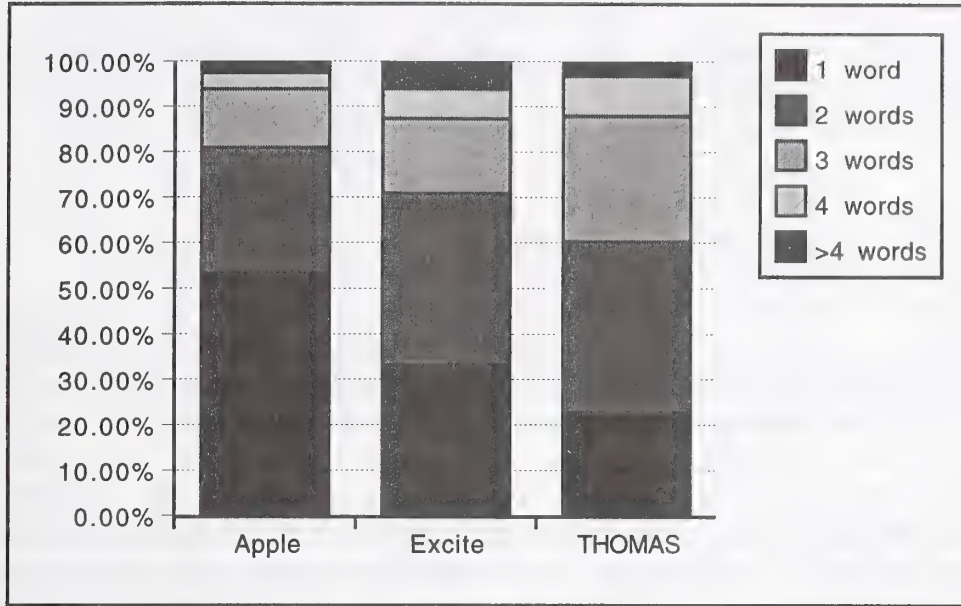


Table 1: Proportion of user queries of different lengths, from three interactive ranked search systems.

irrelevant.

To resolve this conflict, we designed a family of functions that took the raw relevance scores from an arbitrary³ similarity matching function and transformed it into an adjusted score that took coordination level into account. What makes this function unusual is that it is parametrized by the query length, so that for very short queries, the “boost” due to coordination level dominates the raw score, while for longer queries, the coordination boost gradually becomes insignificant.

One function that meets this requirement is:

$$f(s) = \left[s + \frac{(v-1)}{(q-\partial)^2} \right] \cdot \frac{(q-1)}{q}$$

where s is the raw score from the original similarity ranking method, v is the “overlap,” or number of terms in common, between query and document (i.e. the coordination level), q is the query length, and ∂ is a real-valued parameter in the range $[0,1)$ that controls the strength of the coordination effect. We call the function SQR, for Short Query Ranking. (Note that SQR is only used when there are at least two terms in the query.)

While we knew that the SQR approach would address the users’ concerns

³ Our assumption is that the raw function’s range is $[0,1]$.

collection	# terms	R-prec. baseline	R-prec. SQR	change
TREC-4	all	0.2318	0.2501	7.9%
	4	0.1980	0.2216	11.9%
	3	0.1748	0.1929	10.4%
	2	0.1519	0.1843	21.3%

Table 2: *Effect of SQR method on TREC-4 ad hoc queries truncated to various numbers of terms.*

about ranking, we weren't sure whether it might also have a harmful effect on retrieval performance, as measured by traditional tests. It might have been the case that although users preferred the results, the system was retrieving fewer relevant documents. So we designed an experiment to measure the SQR effect. We truncated queries from standard test collections by keeping the n best terms with the highest product of weight and maximum term frequency. Then we ran the system with and without SQR.

The results for the TREC-4 collection are shown in Table 2. We expected that SQR would improve precision for the shortened queries, and it did: 21.3% for the two-word versions. (Note that the precision of the truncated queries will be significantly less than the original queries.) What surprised us was that SQR also improved performance on the original, full-length queries (which averaged 16 words in length) by a modest amount as well, 7.9%.

3. The Road to TREC

When we entered TREC-5, our focus was on the interactive track. We intended to experiment with various interaction capabilities that would help users explore the information space more effectively, including some we had investigated earlier in other contexts [Rose & Belew, 1991; Rose et al., 1993]. We did not expect to score that well at the ad hoc task, since our engine was much lighter weight than most of the other entrants and was designed and optimized for interactive use. However, as first time participants, we underestimated the time it would take to build tools for analyzing the collection. Ultimately we were unable to participate in the interactive track.

Despite this, we were pleased with the initial results we obtained from testing the system on the TREC-4 ad hoc collection. After some tuning, we were achieving what we considered to be good performance levels, as shown in Table 3.

Query construction	Baseline		Feedback		Change	
	Avg. prec.	R-prec.	Avg. prec.	R-prec.	Avg. prec.	R-prec.
automatic	0.2101	0.2692	0.2667	0.3084	26.9%	14.6%
manual	0.2643	0.3155	0.2701	0.3156	2.2%	0.0%

Table 3: Results of running V-Twin on the TREC-4 collection.

	Text size	# Documents	# Terms	Index size	Overhead	Index time
TIPSTER Disk 2	864 MB	231,219	411,864	199 MB	23.0%	11:14:37
TREC Disk 4	1194 MB	293,958	748,921	263 MB	22.0%	17:24:44
TOTAL	2058 MB	525,177		462 MB	22.4%	28:39:21

Table 4: Some statistics about V-Twin's indexes.

We noted that the use of automatic relevance feedback for query expansion had a striking effect on performance, producing average precision levels for fully automatic runs that surpassed our (admittedly inexperienced) manually constructed queries.

V-Twin supports both multiple index searching (with accurate combination of relevance scores incorporating corpus-wide statistics) and index merging. We ran the TREC-4 tests both with a single index and with two separate indexes, one for each disk. Although the merged index was slightly smaller (since nearly 20% of the terms were found in both subcollection indexes), there was no other advantage.

4. TREC-5 Performance

Based on our experience with the TREC-4 data, we felt that V-Twin would perform reasonably well on the TREC-5 ad hoc task. We indexed each disk separately and searched both indexes simultaneously. Some index statistics are shown in Table 4.

We entered two automatic runs. The first, vtwnA1, used just the topic descriptions; the second, vtwnB1, used the entire topic. (We refer to these as the "short query" and "full query" runs.) Both runs used automatic relevance feedback on a preliminary search to generate the final list of matching documents. Both used a stop word list (a slightly modified version of the SMART list), and a heuristic stemmer.

Although the TREC-5 task proved to be more difficult than TREC-4's, we were very pleased with V-Twin's performance, especially considering that this was our first participation in TREC. As shown in Table 5, our overall average precision was 0.1894 for the shorter queries (topic descriptions only) and 0.2065 for the longer queries. Despite their similar average, the two runs were

	All 50 topics		47 topics with > 2 rel. docs	
	Avg. Prec.	R-Prec.	Avg. Prec.	R-Prec.
Short queries	0.1894	0.2315	0.2002	0.2463
Full queries	0.2065	0.2268	0.2189	0.2413

Table 5: Results of two V-Twin runs on the TREC-5 ad hoc task. All queries were automatically constructed.

quite different. On the shorter queries, V-Twin produced better-than-median responses on 30 of the 50 queries, while our response on the longer queries — while worse than the median on 27 of the queries — produced the best average precision of all systems on 4 of the queries.

Our results were especially satisfying to us due to the very low resource requirements of our system in comparison to most other TREC participants. We initially tested indexing on a low-end desktop computer (a Power Macintosh 7200/90, which features a 90 MHz PowerPC 601 processor) with 16 MB of memory, just to demonstrate the scalability of V-Twin. Our final indexing runs were conducted on what would now be considered a mid-performance desktop system (a Power Macintosh 9500/120 with a 120 MHz PowerPC 604); 35 MB of memory (out of a total of 48) were available to the indexing process. In contrast, most of the other entrants used upwards of 128 MB of RAM, with some much higher. Similarly, our index occupied just 462 MB of disk space, while others used as much as 2 GB.

5. Factors Affecting Performance

5.1 Examining Individual Results

By looking at the actual documents retrieved for queries on which we did poorly, we can identify certain areas of the system that need improvement. In certain cases, overly aggressive stemming and case normalization appears to have hurt performance. V-Twin does not contain a built-in stemmer or tokenizer; it provides an abstract class for text translation that allows applications to generate and modify tokens any way they please. For the purposes of TREC-5, we were using a Porter-style heuristic stemmer that predated the rest of the system by several years.

An example of the tokenization and stemming problem can be found by examining the topic at which we did worst (as measured by difference in average precision from median) on the short query run, topic 262. This topic discussed seasonal affective disorder, or SAD. The tokenizer we were using transformed "SAD" to "sad," while the stemmer turned "affective" into "affect" and "seasonal" into "season." As a result, what should have been the highest-weighted terms (based on an IDF weighting) were in fact among the

lowest. When the full queries are used, giving contextual clues like “darkness” and “daylight,” our system performs quite well. Prior to these experiments, we were already developing a less aggressive stemmer; these results seem to confirm the wisdom of that approach. (It is also interesting to note that one of the four relevant documents for this query, AP881125-0098, is a digest of two stories that contain just one occurrence of each of the terms in “seasonal affective disorder,” while containing several mentions of “llama pregnancy.”)

Perhaps the most interesting failures are those catastrophic examples where there were many relevant documents, yet we failed to retrieve any of them. These seem to argue against the use of automatic relevance feedback. One example of these was topic 269, “Define instances of the use of foreign trade as an instrument to achieve national and foreign policy objectives.” Unfortunately, the stemmed forms “define,” “instance,” and “object” were among the highly weighted terms, and these three terms co-occur much more frequently, and with more focused content, in an entirely different context: object-oriented programming. Thus V-Twin found several good matches about such things as the Common Lisp Object System, and then proceeded, through the diligent use of relevance feedback, to eliminate almost anything but these from contention. (The word “foreign” was not sufficiently distinguishing, perhaps because programming environments often have “foreign function” capabilities.) The narrative added similarly generic terms such as “goal” and “order,” so the full query version performed no better.

A second example is topic 297, “Right To Die - Pros and Cons.” Here, three of the query terms in the topic description are stop words, leaving just “pro,” “con,” and “die.” V-Twin’s best matches included a list of addresses from the Federal Register (FR940527-0-00146) that included the abbreviation “CONS” repeated many times, a brief resolution from the Congressional Record (CR93H-16258) in which the speaker **pro** tempore reports on resolution S. **Con.** Res. 33. about adjourning “sine **die**,” and election results from Britain (FT922-13588), with party affiliations (such as Lab for Labour and Con for Conservative) listed after each of a few dozen parliamentary seats. It’s hard to imagine how to solve this problem in an automatic system as long as “right” is a stop word. (The full query version, containing the term “euthanasia,” had no such problems.)

5.2 When Query Expansion Hurt

For both short and long versions of the topics, we used automatic relevance feedback to expand the queries. As shown in Table 6, this had a significant positive effect on performance. However, as noted above, we suspected that relevance feedback was actually causing problems on some of the topics. To test this hypothesis, we examined the effect of feedback by individual topic. We found that in 18 of the 50 topics, feedback was harming performance.

topics	Baseline		Feedback		Change	
	Avg. prec.	R-prec.	Avg. prec.	R-prec.	Avg. prec.	R-prec.
short	0.1584	0.2009	0.1864	0.2220	17.7%	10.5%
long	0.1901	0.2235	0.2065	0.2268	8.6%	1.5%

Table 6: Effect of query expansion by automatic relevance feedback.

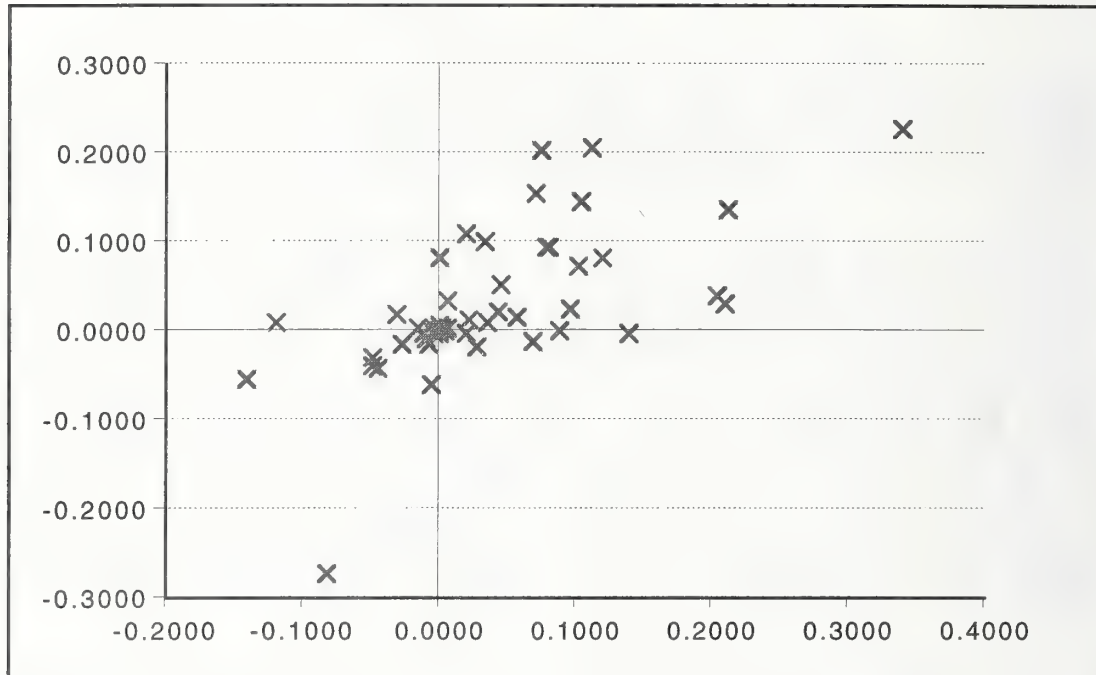


Figure 1: This scatterplot shows the correlation between each query's deviation from the median for all groups (on the X axis) and the affect of relevance feedback on its score (on the Y axis).

Furthermore, in 13 of the 17 short query topics where we did worse than the median (including the three topics discussed in the previous section), feedback had a negative effect.

Figure 1 shows a scatterplot where each query is represented by how our score compared with the median for all groups, and how feedback affected the score. There is a significant correlation between these two measures ($r = 0.64$). This suggests a promising line of inquiry for future experiments: if we can assess the quality of the initial retrieval, we may be able to "cut our losses" rather than compounding the problem by using inappropriate documents for feedback.

5.3 When Stemming Hurt

We performed a similar analysis to assess whether stemming and stop word removal were hurting performance in certain queries. However, the actual

topics	No stemming		Stemming		Change	
	Avg. prec.	R-prec.	Avg. prec.	R-prec.	Avg. prec.	R-prec.
short	0.1352	0.1690	0.1864	0.2220	37.9%	31.4%
long	0.1537	0.1748	0.2065	0.2268	34.4%	29.7%

Table 7: Effect of stemming & stop word elimination.

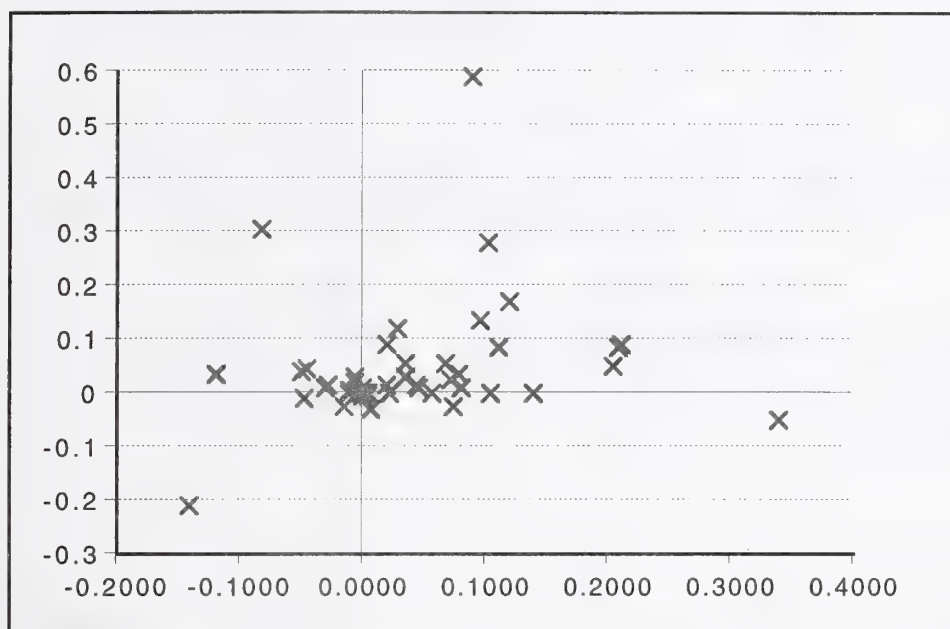


Figure 2: Scatterplot showing each query's deviation from the median for all groups (on the X axis) and the effect of stemming on that query (on the Y axis).

comparison performed was between two different text processing modules which had slightly different tokenizers in addition to differing by the presence or absence of stemming and stop word removal. For convenience, we will refer to the more powerful text processing module as "the stemmer."

As Table 7 indicates, the stemmer was much more effective than the simpler module. Given the uncertainty in the literature about the effects of stemming on performance, we are withholding judgment on these results until we can isolate stemming as a factor.

In any case, we then examined how the stemmer affected individual queries. Again, we found that stemming actually harmed performance on 13 queries. As we predicted, one of these was topic 262 (about Seasonal Affective Disorder), where the stemmed query performed 78% worse than the unstemmed query. However, there was little correlation ($r = 0.20$) in general between topics where stemming hurt and topics where we did poorly, as shown in Figure 2. Only three of the topics on which we scored below the median had a negative effect due to stemming.

# terms	Baseline		SQR		Change	
	Avg. prec.	R-prec.	Avg. prec.	R-prec.	Avg. prec.	R-prec.
all	0.1584	0.2009	0.1595	0.2013	0.7%	0.2%
4	0.1371	0.1801	0.1388	0.1812	1.2%	0.6%
3	0.1277	0.1575	0.1303	0.1582	2.0%	0.4%
2	0.1181	0.1449	0.1224	0.1482	3.6%	2.3%

Table 8: *Effect of SQR on TREC-5 short queries, automatically truncated to various lengths. (No relevance feedback for query expansion.)*

# terms	Baseline		SQR		Change	
	Avg. prec.	R-prec.	Avg. prec.	R-prec.	Avg. prec.	R-prec.
all	0.1864	0.2220	0.1890	0.2322	1.4%	4.6%
4	0.1423	0.1683	0.1566	0.1839	10.0%	9.3%
3	0.1389	0.1645	0.1398	0.1752	0.6%	6.5%
2	0.1198	0.1484	0.1210	0.1521	1.0%	2.5%

Table 9: *Effect of SQR technique on TREC-5 short queries, automatically truncated to various lengths and run with relevance feedback for query expansion.*

6. SQR Revisited

We discovered after submitting our official runs that, due to an error in one of our configuration files, we had not used the SQR technique in the official runs. Because of the large amount of tuning we had done to improve performance, we discovered that the SQR adjustment would have had a relatively insignificant impact on the final results.

We repeated the original SQR experiment described earlier, this time with the TREC-5 corpus. That is, we automatically truncated the queries to two, three, and four words and measured the effect of the SQR technique. In addition, we constructed two-, three-, and four-word manual queries. These results are shown in Tables 8 and 9. As with our previous SQR experiment summarized in Table 2, we found again that SQR did not have a detrimental effect. In fact, it again improved performance, with the largest gains generally coming, as expected, on the most truncated (i.e. shortest) queries.⁴ In contrast to the previous experiment, however, we found this time that the overall performance effect was almost insignificant.

At first, we believed this might be due to the fact that the queries were artificially truncated in the SQR tests. To test this hypothesis, we manually

⁴ The exception, the 4-term test in Table 9, can be explained by the fact that query expansion had a negative effect on this condition; R-precision for the expanded 4-term query was actually worse than R-precision for the unexpanded one.

# terms	Baseline		SQR		Change	
	Avg. prec.	R-prec.	Avg. prec.	R-prec.	Avg. prec.	R-prec.
4	0.1630	0.2113	0.1642	0.2120	0.7%	0.3%
3	0.1597	0.1947	0.1623	0.1964	1.6%	0.9%
2	0.1296	0.1750	0.1332	0.1761	2.8%	0.6%

Table 10: *Effect of SQR technique on manually truncated TREC-5 short queries, run without query expansion.*

generated an alternate set of 2-, 3-, and 4-word queries. As shown in Table 10, the basic results are the same: While overall performance of manually truncated queries is better than that of automatically truncated queries, the SQR technique has only a small effect on the result.

What might account for such a dramatic difference between the earlier experiment and the later ones in the effect of the SQR method? We believe the primary explanation lies in the low baseline we used when performing our original SQR experiment on TREC-4 data. This baseline predated our participation in TREC and used less effective term weighting and normalization algorithms. Thus our current assessment of the SQR method is that first, it appears to solve a usability problem without harming retrieval performance, and second, in cases of low absolute performance it may have a significant positive effect. To put it another way, the SQR technique can be used to partially compensate for weaknesses in a retrieval system's ranking algorithm.

7. Conclusions

Our work in TREC-5 has suggested several promising areas on which we may focus in the future, such as ways to detect when relevance feedback will harm performance. We have also learned a great deal about tuning our engine for the TREC task. In particular, we now believe that we may have overfit to the TREC-4 data when preparing for TREC-5.

We found further evidence that our SQR technique, which solves a problem with users' perception of ranking, has no detrimental effect on performance, and in fact seems to improve precision slightly. The results also reaffirmed our belief that searching is most effective when treated as an interactive process rather than a batch event. Perhaps most importantly, we successfully demonstrated that a lightweight engine such as V-Twin could effectively handle a larger scale search task, outperforming systems with much greater disk and memory requirements.

References

- Rose, D. E. and Belew, R. K. (1991). "Toward a Direct-Manipulation Interface for Conceptual Information Retrieval Systems." In Martin Dillon, ed., *Interfaces for Information Retrieval and Online Systems: The State of the Art*, pp. 39-54, Greenwood Press, Westport, CT.
- Rose, D. E., Mander, R., Oren, T., Ponceleón, D. B., Salomon, G., Wong, Y. (1993). "Content Awareness in a File System Interface: Implementing the 'Pile' Metaphor for Organizing Information. *SIGIR-93*, pp. 260-269.
- Rose, D. E. and Cutting, D. R. (1996). "Ranking for Usability: Enhanced Retrieval for Short Queries." Apple Technical Report #163.

NATURAL LANGUAGE INFORMATION RETRIEVAL: TREC-5 REPORT

**Tomek Strzalkowski¹ Louise Guthrie² Jussi Karlgren³ Jim Leistensnider²
Fang Lin¹ Jose Perez-Carballo⁴ Troy Straszheim³ Jin Wang¹ Jon Wilding²**

¹GE Corporate Research & Development

²Lockheed Martin Corporation

³Department of Computer Science, New York University

⁴School of Communication, Information and Library Studies, Rutgers University

ABSTRACT

In this paper we report on the joint GE/Lockheed Martin/Rutgers/NYU natural language information retrieval project as related to the 5th Text Retrieval Conference (TREC-5). The main thrust of this project is to use natural language processing techniques to enhance the effectiveness of full-text document retrieval. Since our first TREC entry in 1992 (as NYU team) the basic premise of our research was to demonstrate that robust if relatively shallow NLP can help to derive a better representation of text documents for statistical search. TREC-5 marks a shift in this approach away from text representation issues and towards query development problems. While our TREC-5 system still performs extensive text processing in order to extract phrasal and other indexing terms, our main focus this year was on query construction using words, sentences, and entire passages to expand initial topic specifications in an attempt to cover their various angles, aspects and contexts. Based on our earlier TREC results indicating that NLP is more effective when long, descriptive queries are used, we allowed for liberal expansion with long passages from related documents imported verbatim into the queries. This method appears to have produced a dramatic improvement in the performance of two different statistical search engines that we tested (Cornell's SMART and NIST's Prise) boosting the average precision by at least 40%.

The overall architecture of TREC-5 system has also changed in a number of ways from TREC-4. The most notable new feature is the stream architecture in which several independent, parallel indexes are built for a given collection, each index reflecting a different representation strategy for text documents. Stream indexes are built using a mixture of different indexing approaches, term extracting, and weighting strategies. We used both SMART and Prise base indexing engines, and selected optimal term weighting strategies for each stream, based on a training collection of approximately 500 MBytes. The final results are produced by a merging procedure that combines ranked list of documents obtained by searching all stream indexes with appropriately preprocessed queries. This allows for an effective combination of alternative retrieval and filtering methods, creating into a meta-search where the contribution of each stream can be optimized through training.

1.0 INTRODUCTION AND BACKGROUND

A typical (full-text) information retrieval (IR) task is to select documents from a database in response to a user's query, and rank these documents according to relevance. This has been usually accomplished using statistical methods (often coupled with manual encoding) that (a) select terms (words, phrases, and other units) from documents that are deemed to best represent their content, and (b) create an inverted index file (or files) that provide an easy access to documents containing these terms. A subsequent search process will attempt to match preprocessed user queries against term-based representations of documents in each case determining a degree of relevance between the two which depends upon the number and types of matching terms. Although many sophisticated search and matching

methods are available, the crucial problem remains to be that of an adequate representation of content for both the documents and the queries.

In term-based representation, a document (as well as a query) is transformed into a collection of weighted terms, derived directly from the document text or indirectly through thesauri or domain maps. The representation is anchored on these terms, and thus their careful selection is critical. Since each unique term can be thought to add a new dimensionality to the representation, it is equally critical to weigh them properly against one another so that the document is placed at the correct position in the N-dimensional term space. Our goal here is to have the documents on the same topic placed close together, while those on different topics placed sufficiently apart. Unfortunately, we often do not know how to compute terms weights. The statistical weighting formulas, based on terms distribution within the database, such as $tf \cdot idf$, are far from optimal, and the assumptions of term independence which are routinely made are false in most cases. This situation is even worse when single-word terms are intermixed with phrasal terms and the term independence becomes harder to justify.

The simplest word-based representations of content, while relatively better understood, are usually inadequate since single words are rarely specific enough for accurate discrimination, and their grouping is often accidental. A better method is to identify groups of words that create meaningful phrases, especially if these phrases denote important concepts in the database domain. For example, "joint venture" is an important term in the Wall Street Journal (WSJ henceforth) database, while neither "joint" nor "venture" are important by themselves. In the retrieval experiments with the training TREC database, we noticed that both "joint" and "venture" were dropped from the list of terms by the system because their idf (inverted document frequency) weights were too low. In large databases, such as TIPSTER, the use of phrasal terms is not just desirable, it becomes necessary.

There are a number of ways to obtain "phrases" from text. These include generating simple collocations, statistically validated N-grams, part-of-speech tagged sequences, syntactic structures, and even semantic concepts. Some of these techniques are aimed primarily at identifying multi-word terms that have come to function like ordinary words, for example "white collar" or "electric car", and capturing other co-occurrence idiosyncrasies associated with certain types of texts. This simple approach has proven quite effective for some systems, for example the Cornell group reported (Buckley, 1995) that adding simple collocations to the list of available terms can increase retrieval precision by as much as 10%.

Other more advanced techniques of phrase extraction, including extended N-grams and syntactic parsing, attempt to uncover "concepts", which would capture underlying semantic uniformity across various surface forms of expression. Syntactic phrases, for example, appear reasonable indicators of content, arguably better than proximity-based phrases, since they can adequately deal with word order changes and other structural variations (e.g., "college junior" vs. "junior in college" vs. "junior college"). A subsequent regularization process, where alternative structures are reduced to a "normal form", helps to achieve a desired uniformity, for example, "college+junior" will represent a college for juniors, while "junior+college" will represent a junior in a college. A more radical normalization would have also "verb object", "noun rel-clause", etc. converted into collections of such ordered pairs. This head+modifier normalization has been used in our system, and is further described in section 3. It has to be noted here that the use of full-scale syntactic analysis is severely pushing the limits of practicality of an information retrieval system because of the increased demand for computing power and storage. At the same time, while the gain in recall and precision has not been negligible, no dramatic breakthrough has occurred either.

This state of affairs has prompted us take a closer look at the phrase selection and representation process. In TREC-3 we showed that an improved weighting scheme for compound terms, including phrases and proper names, leads to an overall gain in retrieval accuracy. The fundamental problem, however, remained to be the system's inability to recognize, in the documents searched, the presence or absence of the concepts or topics that the query is asking for. The main reason for this was, we noted, the limited amount of information that the queries could convey on various aspects of topics they represent. Therefore, starting with TREC-4, and continuing on a much larger scale in TREC-5, we started experimenting with manual and automatic query building techniques. The purpose was to devise a method for full-text query expansion that would allow for creating exhaustive search queries such that: (1) the performance of any system using these queries would be significantly better than when the system is run using the original topics, and (2) the method could be eventually automated or semi-automated so as to be useful to a non-expert user. Our preliminary results from TREC-5 evaluations show that this approach is indeed very effective.

In this paper we describe the overall organization of our TREC-5 system, and then discuss the official and “unofficial” experiments and their results, as well as our future research plans.

2.0 STREAM-BASED INFORMATION RETRIEVAL MODEL

Our NLIR system encompasses several statistical and natural language processing (NLP) techniques for robust text analysis. These has been organized together into a “stream model” in which alternative methods of document indexing are strung together to perform in parallel. Stream indexes are built using a mixture of different indexing approaches, term extracting and weighting strategies, even different search engines. The final results are produced by merging ranked lists of documents obtained from searching all stream indexes with appropriately preprocessed queries, i.e., phrases for phrase stream, names for names stream, etc. The merging process weights contributions from each stream using a combination that was found the most effective in training runs. This allows for an easy combination of alternative retrieval and routing methods, creating a meta-search strategy which maximizes the contribution of each stream. The stream model is illustrated in Figure 1. We used both Cornell’s SMART version 11, and NIST’s Prize search engines.

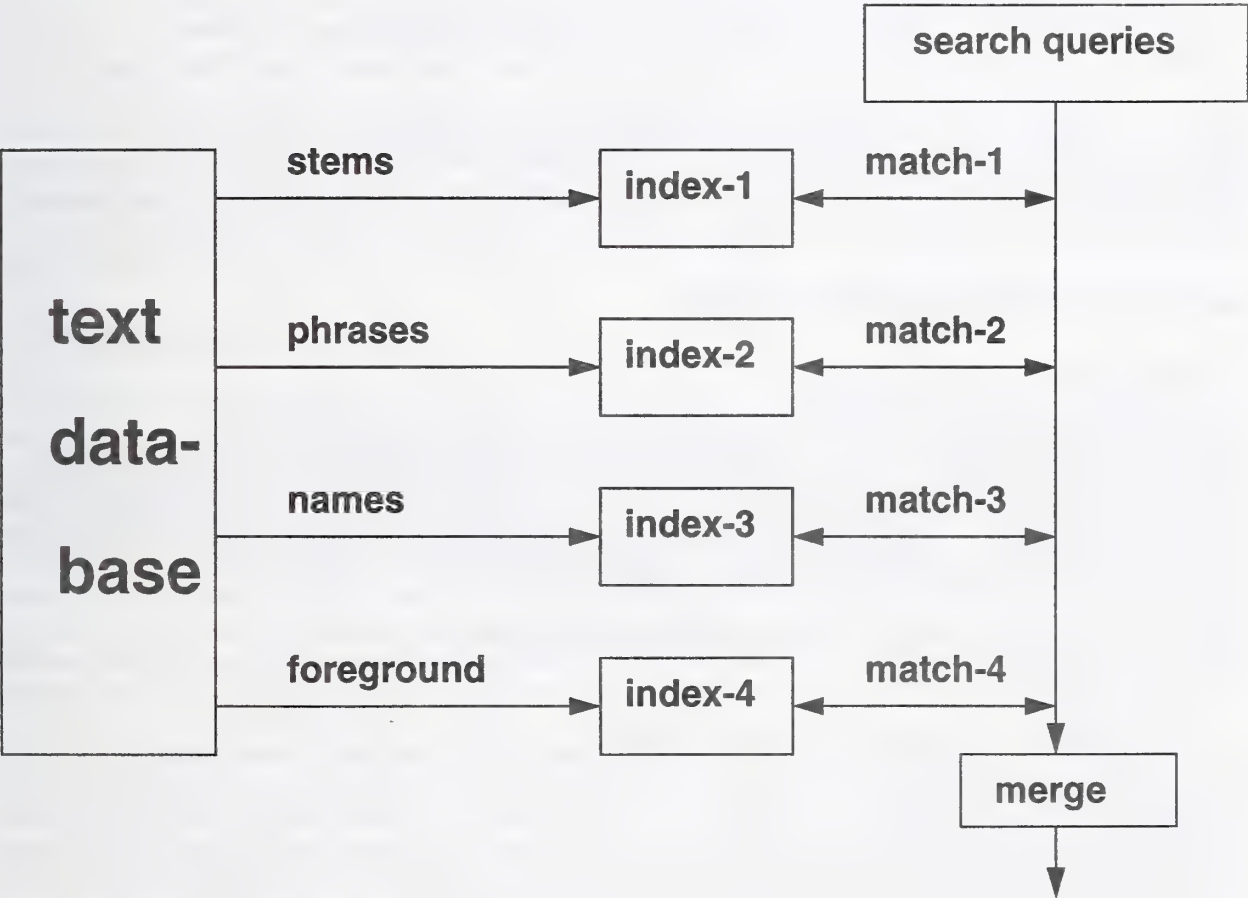


FIGURE 1. Stream organization concept.

Our TREC-5 system employs a suite of advanced natural language processing techniques in order to assist the statistical retrieval engine in selecting appropriate indexing terms for documents at hand, and to assign them semantically validated weights. The following term extraction methods has been used, which correspond to some of the streams we used:

1. Eliminate stopwords: original text words minus certain no-content words are used to index documents.

2. Morphological stemming: we normalize across morphological word variants (e.g., “proliferation”, “proliferate”, “proliferating”) using a lexicon-based stemmer.
3. Phrase extraction: we use various shallow text processing techniques, such as part-of-speech tagging, phrase boundary detection, and word co-occurrence metrics to identify stable strings of words, such as “joint venture”.
4. Phrase normalization: we identify “head+modifier” pairs in order to normalize across syntactic variants such as “weapon proliferation”, “proliferation of weapons”, “proliferate weapons”, etc. into “weapon+proliferate”.
5. Proper names: we identify proper names for indexing, including people names and titles, location names, organization names, etc.

Among the advantages of the stream architecture we may include the following:

- stream organization makes it easier to compare the contributions of different indexing features or representations. For example, it is easier to design experiments which allow us to decide if a certain representation adds information which is not contributed by other streams.
- it provides a convenient testbed to experiment with algorithms designed to merge the results obtained using different IR engines and/or techniques.
- it becomes easier to fine-tune the system in order to obtain optimum performance
- it allows us to use any combination of Tipster-compliant IR engines without having to modify their code at all.

In addition, several researchers in past TRECs have noticed that different systems may have similar performance but retrieve different documents, thus suggesting that they may complement one another. It has been reported that the use of different sources of evidence increases the performance of a system (see for example, ref Saracevic and Kantor and ref Bruce Croft).

3.0 ADVANCED LINGUISTIC STREAMS

3.1 Head-Modifier Pairs Stream

Our most linguistically advanced stream is the head+modifier pairs stream. In this stream, documents are reduced to collections of word pairs derived via syntactic analysis of text followed by a normalization process intended to capture semantic uniformity across a variety of surface forms, e.g., “information retrieval”, “retrieval of information”, “retrieve more information”, “information that is retrieved”, etc. are all reduced to “retrieve+information” pair, where “retrieve” is a head or operator, and “information” is a modifier or argument.

The pairs stream is derived through a sequence of processing steps that include:

- Part-of-speech tagging
- Lexicon-based word normalization (extended “stemming”)
- Syntactic analysis with TTP parser
- Extraction of head+modifier pairs
- Corpus-based disambiguation of long noun phrases

3.1.1 Part-of-speech tagging

We used a version of Brill’s rule based tagger trained on Wall Street Journal texts to preprocess linguistic streams used by SMART. We also used BBN’s POST tagger as part of our NYU-based Prise system. Both systems use Penn. Treebank Tagset developed at University of Pennsylvania, and have compatible levels of performance.

3.1.2 Lexicon-based word normalization

Word stemming has been an effective way of improving document recall since it reduces words to their common morphological root, thus allowing more successful matches. On the other hand, stemming tends to decrease retrieval precision, if care is not taken to prevent situations where otherwise unrelated words are reduced to the same stem. In our system we replaced a traditional morphological stemmer with a conservative dictionary-assisted suffix trimmer.¹

The suffix trimmer performs essentially two tasks:

1. it reduces inflected word forms to their root forms as specified in the dictionary, and
2. it converts nominalized verb forms (e.g., “implementation”, “storage”) to the root forms of corresponding verbs (i.e., “implement”, “store”).

This is accomplished by removing a standard suffix, e.g., “stor+age”, replacing it with a standard root ending (“+e”), and checking the newly created word against the dictionary, i.e., we check whether the new root (“store”) is indeed a legal word. Below is a small example of text before and after stemming.

While serving in South Vietnam, a number of U.S. Soldiers were reported as having been exposed to the defoliant Agent Orange. The issue is veterans entitlement, or the awarding of monetary compensation and/or medical assistance for physical damages caused by Agent Orange.

serve south vietnam number u.s. soldier expose defoliant agent orange veteran entitle
award monetary compensate medical assist physical damage agent orange

Please note that proper names, such as South Vietnam and Agent Orange are identified separately through the name extraction process described below. Note also that various “stopwords” (e.g., prepositions, conjunctions, articles, etc.) are removed from text.

3.1.3 Syntactic analysis with TTP

TTP (Tagged Text Parser) is based on the Linguistic String Grammar developed by Sager (1981). The parser currently encompasses some 400 grammar productions, but it is by no means complete. The parser’s output is a regularized parse tree representation of each sentence, that is, a representation that reflects the sentence’s logical predicate-argument structure. For example, logical subject and logical object are identified in both passive and active sentences, and noun phrases are organized around their head elements. The parser is equipped with a powerful skip-and-fit recovery mechanism that allows it to operate effectively in the face of ill-formed input or under a severe time pressure. When parsing the TREC-3 collection of more than 500 million words, we found that the parser’s speed averaged between 0.17 and 0.26 seconds per sentence, or up to 80 words per second, on a Sun’s SparcStation10. In addition, TTP has been shown to produce parse structures which are no worse than those generated by full-scale linguistic parsers when compared to hand-coded Treebank parse trees.

TTP is a full grammar parser, and initially, it attempts to generate a complete analysis for each sentence. However, unlike an ordinary parser, it has a built-in timer which regulates the amount of time allowed for parsing any one sentence. If a parse is not returned before the allotted time elapses, the parser enters the skip-and-fit mode in which it will try to “fit” the parse. While in the skip-and-fit mode, the parser will attempt to forcibly reduce incomplete constituents, possibly skipping portions of input in order to restart processing at a next unattempted constituent. In other words, the parser will favor reduction to backtracking while in the skip-and-fit mode. The result of this strategy is an approximate parse, partially fitted using top-down predictions. The fragments skipped in the first pass are not thrown out, instead they are analyzed by a simple phrasal parser that looks for noun phrases and relative clauses and then attaches the recovered material to the main parse structure. Full details of TTP parser have been described in the

1. Dealing with prefixes is a more complicated matter, since they may have quite strong effect upon the meaning of the resulting term, e.g., un- usually introduces explicit negation

TREC-1 report (Strzalkowski, 1993a), as well as in other works (Strzalkowski, 1992; Strzalkowski & Scheyen, 1996).

3.1.4 Extracting head+modifier pairs

Syntactic phrases extracted from TTP parse trees are head-modifier pairs. The head in such a pair is a central element of a phrase (main verb, main noun, etc.), while the modifier is one of the adjunct arguments of the head. It should be noted that the parser's output is a predicate-argument structure centered around main elements of various phrases. The following types of pairs are considered: (1) a head noun and its left adjective or noun adjunct, (2) a head noun and the head of its right adjunct, (3) the main verb of a clause and the head of its object phrase, and (4) the head of the subject phrase and the main verb. These types of pairs account for most of the syntactic variants for relating two words (or simple phrases) into pairs carrying compatible semantic content. This also gives the pair-based representation sufficient flexibility to effectively capture content elements even in complex expressions. There are of course exceptions. For example, the three-word phrase "former Soviet president" has been broken into two pairs "former president" and "Soviet president", both of which denote things that are potentially quite different from what the original phrase refers to, and this fact may have potentially negative effect on retrieval precision. This is one place where a longer phrase appears more appropriate.

3.1.5 Corpus-based disambiguation of long noun phrases

The notorious structural ambiguity of nominal compounds remains a serious difficulty in obtaining quality head-modifier pairs. What it means is that word order information cannot be reliably used to determine relationships between words in complex phrases, which is required to decompose longer phrases into meaningful head+modifier pairs. In order to cope with ambiguity, the pair extractor looks at the distribution statistics of the compound terms to decide whether the association between any two words (nouns and adjectives) in a noun phrase is both syntactically valid and semantically significant. For example, we may accept language+natural and processing+language from "natural language processing" as correct, however, case+trading would make a mediocre term when extracted from "insider trading case". On the other hand, it is important to extract trading+insider to be able to match documents containing phrases "insider trading sanctions act" or "insider trading activity". Phrasal terms are extracted in two phases. In the first phase, only unambiguous head-modifier pairs are generated, while all structurally ambiguous noun phrases are passed to the second phase "as is". In the second phase, the distributional statistics gathered in the first phase are used to predict the strength of alternative modifier-modified links within ambiguous phrases. For example, we may have multiple unambiguous occurrences of "insider trading", while very few of "trading case". At the same time, there are numerous phrases such as "insider trading case", "insider trading legislation", etc., where the pair "insider trading" remains stable while the other elements get changed, and significantly fewer cases where, say, "trading case" is constant and the other words change.

The phrase decomposition procedure is performed after the first phrase extraction pass in which all unambiguous pairs (noun+noun and noun+adjective) and all ambiguous noun phrases are extracted. Any nominal string consisting of three or more words of which at least two are nouns is deemed structurally ambiguous. In the Tipster corpus, about 80% of all ambiguous nominals were of length 3 (usually 2 nouns and an adjective), 19% were of length 4, and only 1% were of length 5 or more. The phrase decomposition algorithm has been described in detail in (Strzalkowski, 1995). The algorithm was shown to provide about 70% recall and 90% precision in extracting correct head+modifier pairs from 3 or more word noun groups in TREC collection texts. In terms of the total number of pairs extracted unambiguously from the parsed text, the disambiguation step recovers an additional 10% to 15% of pairs, all of which would otherwise be either discarded or misrepresented.

3.2 Linguistic Phrase Stream

To test the effectiveness of noun phrases, we choose a stream which utilize simple noun phrases as atomic index terms. The original text is part-of-speech tagged and stemmed. The noun phrases are then identified by regular expression rules on the part-of-speech tags. The major rules are:

1. a sequence of modifiers (vbnlvbgllj) followed by at least one noun, such as: “cryonic suspend”, “air traffic control system”;
2. proper noun(s) modifying a noun, such as: “u.s. citizen”, “china trade”;
3. proper noun(s) (might contain ‘&’), such as: “warren commission”, “national air traffic controller”.

The length of phrases is limited to maximum 7 words.

The Smart retrieval system gives us many choices of weighting schemes. In order to choose an effective weighting scheme, some experiments were carried out. The result suggests that the best weighting for phrases is Inc.lsc (Inc for documents, lsc for queries).

3.3 Name Stream

Proper names, of people, places, events, organizations, etc., are often critical in deciding relevance of a document. Since names are traditionally capitalized in English text, spotting them is relatively easy, most of the time. Many names are composed of more than a single word, in which case all words that make up the name are capitalized, except for prepositions and such, e.g., The United States of America. It is important that all names recognized in text, including those made up of multiple words, e.g., South Africa or Social Security, are represented as tokens, and not broken into single words, e.g., South and Africa, which may turn out to be different names altogether by themselves. On the other hand, we need to make sure that variants of the same name are indeed recognized as such, e.g., U.S. President Bill Clinton and President Clinton, with a degree of confidence. One simple method, which we use in our system, is to represent a compound name dually, as a compound token and as a set of single-word terms. This way, if a corresponding full name variant cannot be found in a document, its component words matches can still add to the document score. A more accurate, but arguably more expensive method would be to use a substring comparison procedure to recognize variants before matching.

In our system names are identified by the parser, and then represented as strings, e.g., south+africa. The name recognition procedure is extremely simple, in fact little more than the scanning of successive words labeled as proper names by the tagger (“np” and “nps” tags). Single-word names are processed just like ordinary words, except for the stemming which is not applied to them. We also made no effort to assign names to categories, e.g., people, companies, places, etc., a classification which is useful for certain types of queries (e.g., “To be relevant a document must identify a specific generic drug company”). In the TREC-5 database, compound names make up about 8% of all terms generated. A small sample of compound names extracted is listed below:

```
right+wing+christian+fundamentalism
u.s+constitution
gun+control+legislation
national+railroad+transportation+corporation
superfund+hazardous+waste+cleanup+programme
u.s+government
united+states
exxon+valdez
dow_corning+corporation
chairman+julius+d+winer
new+york
wall+street+journal
```


4.0 OTHER STREAMS

4.1 Stems Stream

The stems stream is the simplest, yet, it turns out, the most effective of all streams, a backbone in our multi-stream model. It consists of stemmed non-stop single-word tokens (plus hyphenated phrases). Our early experiments with multi-stream indexing using SMART suggested that the most effective weighting of this stream is *Inc.ltc*, which yields the best average precision, whereas *Inc.ntc* slightly sacrifices the average precision, but gives better recall.

4.2 Unstemmed Word Stream

For the routing experiment with PRISE we used also a plain text stream. This stream was obtained by indexing the text of the documents "as is" without stemming or any other processing and running the unprocessed text of the queries against that index.

4.3 Fragment Stream

For the routing experiments with PRISE we also used a stream of fragments. This was the result of splitting the documents of the STEM stream into fragments of constant length (1024 characters) and indexing each fragment as if it were a different document. The queries used with this stream were the usual stem queries. For each query, the resulting ranking was filtered to keep, for each document, the highest score obtained by the fragments of that document.

5.0 MERGING STRATEGY

The results obtained from different streams, i.e., ranked lists of documents retrieved from searching each stream, were merged into a single final ranking. The final score of each document is computed by combining the relevance estimate given in the scores of all the streams where it is retrieved. The merge is based on two factors:

1. document's relevance estimates from various streams;
2. the overall retrieval effectiveness of those streams in general.

A more effective stream will carry a higher weight, and a higher ranking in that stream will have a larger effect to move the document up in the merged ranking. The entire merging procedure is carried out in two steps. The first step is same-system inter-stream merge, in which the rankings obtained from the streams using the same retrieval system (e.g., SMART) are combined together. The second step is the inter-system merge. In each case a different merging algorithm is used.

5.1 Inter-stream merging in SMART

Each stream carries some unique type of information about the documents it indexes. It is therefore critical that the merging process knows how to properly combine and translate the stream-level rankings into one global ranking.

We used the following two principal sources of information about each stream to weigh their relative contributions to the final ranking:

- an actual ranking obtained from a training run (training data, old queries);
- an estimated retrieval precision at certain range of ranks.

This estimate varies from stream to stream. For example, an estimate of the stems stream may consist of the following (see also Table 1): the precision within top 10 is 39%, between the 11th to the 20th is 34%, etc.

TABLE 1. Precision distribution

RANK	PRECISION
1-10	0.39
11-20	0.34
31-50	0.28
51-100	0.21
101-200	0.13

The final rank of a document (d) is defined as:

$$\sum_{i=1..4} \{ A[i] \times (\text{Score}[i](d) \times \text{Prec}[i](\text{rank})) \}$$

where $i = 1..4$ stands for the four different streams we used; $A[i]$ is the weight for each stream which is acquired through experiments; $\text{Score}[i](d)$ is the normalized score of the document against the query within the stream i ; $\text{Prec}[i](\text{rank})$ is the estimate precision described above.

5.2 Inter-stream merging in PRISE

There are many ways in which the information obtained from different streams can be merged in order to obtain the final results. In our experiments with PRISE we tried several methods and chose the ones that seemed to be the best for our official results. The experiments that helped us chose the best merging technique were performed using a 500-MBytes dry-run collection that was created using past TRECs data for which we already had relevance information available. Some of the methods we tried include:

- linear combinations: i.e. multiply the score obtained by each stream by a constant and then add the score of all streams together.
- change the scores of the documents in order to push to the top of the ranking all documents that appeared in more than one stream. The first group of documents of the resulting ranking appeared in n streams, the second group in $n-1$, etc. At the bottom of this ranking would be the documents that appeared in only one stream.
- a combination of the previous two: multiply each stream by a different constant (determined by experiment) and add all streams together. Then for each document multiply the score by a number which is a function of the number of streams in which that document appeared.

In the dry run experiments that we performed with PRISE the third method achieved the highest performance. Using this method we obtained increases in performance of around 40% over the performance of the stem stream alone. The assumption that supports the use of the third method is that each additional stream in which a document appears adds to the evidence that the document may be relevant. The function that we used to multiply the score of each document (which was a linear combination of the scores of all the streams in which it appeared) was:

$$(0.9 + \text{number-of-streams}/10)$$

where *number-of-streams* is the number of streams in which the document appears. Thus, if the document appears in only one ranking the score is not changed, if it appears in 2 rankings the new score is 1.1 times the old score, etc.

5.3 Inter-system merging

The same merging technique can be also used to merge results from two different retrieval systems. Conceivably the less similarity two systems have, the better result can be expected from the merge. The reason is that dissimilar systems tend to make their decisions based on different document and query features. Incorporating more information is the key for the merging to be successful. Nonetheless, even with two rather similar systems, Smart and PriSe, we still see 10-20% improvement in the runs where we have used the inter-system merging.

6.0 SMART AND PRISE

6.1 SMART system as used in TREC-5

We used Smart system V.11 available from Cornell University. In some runs, including NLP Track evaluations **sbase1**, **sbase2**, **genlp2**, **genlp3**, SMART own text pre-processing facilities were used: stopword filtering, suffix stemming, proper noun detection, statistical phrases grouping using high frequency adjacent word pairs (bigrams). In all other runs where the multi-stream model was utilized, these simple techniques are replaced by more advanced linguistic processes that include lexicon-based morphological normalization, proper names recognition, syntactic phrase extraction and so forth.

6.2 PRISE system as used in TREC-5

PriSe is a statistical information retrieval system developed by Donna Harman at NIST. The system has been unchanged since TREC-4. For details, the reader is referred to our earlier TREC-based publications (e.g., Strzalkowski, 1994), or (Harman&Candela, 1991).

7.0 TREC-5 AD-HOC RUNS

7.1 Automatic Ad-Hoc Runs

7.1.1 Ad-hoc experiments using PRISE

For the automatic ad-hoc experiment with PRISE we used the following streams, along with the following merging coefficients:

Coeff	Stream
1	<i>stems</i>
4	<i>locality with n=20</i>
1	<i>fragments</i>
1	<i>words</i>
3	<i>pairs</i>
1	<i>names</i>

There was not enough time to run experiments involving many other possible combinations. The numbers listed above were determined through limited experiments using the dry-run collection. There was no time at all to run experiments for the fragments and plain-text streams.

The scoring function used with both the ad-hoc and routing experiments is as follows:

$$final-score(d) = score(d) * (0.9 + number-of-streams(d)/10)$$

where number-of-streams is the number of streams in which document *d* is retrieved.

7.1.2 Ad-hoc experiments using SMART

For the automatic ad-hoc experiment with PRISE we used the following streams, along with the following merging coefficients:

Coeff	Stream
4	<i>stems</i>
3	<i>phrases</i>
3	<i>pairs</i>
1	<i>names</i>

The scoring function is discussed in detail in section 6.2

7.2 Query Expansion Experiments in Manual Runs

The purpose of query expansion in information retrieval is to make the user query resemble more closely the documents it is expected to retrieve. In a typical situation, content terms from documents judged relevant documents are added to the query, and other terms weights are adjusted in order to reflect this new evidence. This process can be performed automatically using a relevance feedback method, e.g., Roccio's, or it can be done manually by the user. A serious problem with the content-term expansion is its limited ability to capture and represent many important aspects of what makes some documents relevant to the query, including particular term co-occurrence patterns, and other hard-to-measure text features, such as discourse structure. Additionally, it depends on the partial relevance information, which is normally unavailable, or unreliable.

An alternative to term-only expansion is a full-text expansion which we tried for the first time this year in TREC-5. In our approach, queries are expanded by pasting in entire sentences, paragraphs, and other sequences directly from ANY document. To make this process efficient, we first perform a search with the original, un-expanded queries, and then use top N (10, 20) returned documents for query expansion. These documents are not judged on relevancy nor assumed relevant, instead, they are scanned for passages that contain a concept referred to in the query. Subject to some further "fitness criteria", these passages are then imported verbatim into the query. This can be accomplished manually, as we did in TREC-5 main adhoc manual runs, or automatically, as we tried in one of the NLP Track runs. The resulting expanded queries undergo the usual text processing steps, before the search is run again.

The initial evaluations indicate that queries expanded this way are improving the system's performance (precision and recall) by as much as 40%. At this time, automatic text expansion produces less effective queries than manual expansion, primarily due to a relatively unsophisticated mechanism used to identify concepts in the queries (see section 10.1 for details).

7.2.1 Query expansion guidelines

We have adopted the following guidelines for query expansion. They were constructed to observe realistic limits of the manual process, and to prepare ground for eventual automation.

1. SMART and Prise are run using the original queries, with all streams, and our regular merge.
2. Users (i.e., team members) get top 10 docs retrieved by each of "their" queries (we used 5 to 10 queries per person, taking advantage of our team size).
3. Each query is manually expanded using phrases, sentences, etc. found in any of the top 10 documents for this query. Text can both added and deleted, though care is taken to assure that the final query has the same format as the original, and that all expressions added are well-formed English strings (though not necessarily sentences) ended with a period. **A limit of 30 minutes per query in a single block of time is observed.**

4. Expanded queries are sent for NL processing, then run through all streams and search engines as in step 1. This constitutes our genrl3 run.²
5. Queries, and new top 10 documents are returned to their “expanders” who are now asked to judge relevance.³
6. A relevance feedback is run based on top 10 relevant/non-relevant info. This constitutes our genrl4 run.

The actual time table used to prepare TREC-5 manual run is given below:

July 08, noon: queries and top docs distributed
 July 10, noon: first round expansion due
 July 10, 4 pm: extended queries NLP processed
 July 11, noon: extended queries and new top 10 docs distributed
 July 11, 4 pm: additional query revisions, if any, due
 July 11, 6 pm: all revised queries NLP processed
 July 12, noon: new top 10 docs redistributed for revised queries
 July 12, 3 pm: relevant/non-relevant judgements on final top 10 due

8.0 TREC-5 ROUTING RUNS

Routing is a process in which a stream of previously unseen documents are filtered and distributed among a number of standing profiles, also known as routing queries. In routing, documents can be assigned to multiple profiles. In categorization, a type of routing, a single best matching profile is selected for each document. Routing is harder to evaluate in a standardized setup than the retroactive retrieval because of its dynamic nature, therefore a simulated routing mode has been used in TREC. A simulated routing mode (TREC-style) means that all routing documents are available at once, but the routing queries (i.e., terms and their weights) are derived with respect to a different training database, specifically TREC collections from previous evaluations. This way, no statistical or other collection-specific information about the routing documents is used in building the profiles, and the participating systems are forced to make assumptions about the routing documents just like they would in real routing. However, no real routing occurs, and the prepared routing queries are run against the routing database much the same way they would be in an ad-hoc retrieval. Documents retrieved by each routing query, ranked in order of relevance, become the content of its routing bin.

8.1 Standard Routing

8.1.1 Routing experiments using PRISE

For the standard routing experiment using PRISE we used the following streams and coefficients:

Coeff	Stream
1	<i>stems</i>
4	<i>locality with n=20</i>
3	<i>pairs</i>
1	<i>names</i>

-
2. A few queries were “corrected” to fix formatting problems. Other changes were allowed if expanders felt “unhappy” with their queries.
 3. For some queries, no relevant documents were found in top 10. These queries were further expanded and rerun one more time.

The value of the coefficients was determined using the dry-run collection.

The same scoring function was used with the ad-hoc and routing experiments:

$$final_score(d) = score(d) * (.9 + number-of-streams(d)/10)$$

where number-of-streams is the number of streams in which document d is retrieved.

8.1.2 Routing experiments using SMART

In Smart routing, automatic relevance feedback was performed to build routing queries using the training data available from previous TRECs. The routing queries, split into streams, were then run against stream-indexed routing collection. The weighting scheme was selected in such a way that no collection-specific information about the current routing data has been used. Instead, collection-wide statistics, such as idf weights, were those derived from the training data. The routing was carried out in the following four steps:

1. A subset of the previous TREC collections was chosen as the training set, and four index streams were built. Queries were also processed and run against the indexes. For each query, 1000 documents are retrieved. The weighting schemes used were: Inc.ltc for stems, ltc.ntc for phrases, ltc.ntc for head+modifier pairs, and ltc.ntc for names.
2. The final query vector was then updated through an automatic feedback step using the known relevance judgments. Up to 350 terms occurring in the most relevant documents were added to each query. Two alternative expanded vectors were generated for each query using different sets of Rocchio parameters.
3. For each query, the best performing expansion was retained. These were submitted to NIST as official routing queries.
4. The final queries were run against the four-stream routing test collection and retrieved results were merged.

For the routing runs with SMART we used the following streams, along with the merge coefficients:

Coeff	Stream
4	<i>stems</i>
3	<i>phrases</i>
3	<i>pairs</i>
1	<i>names</i>

8.2 Classification-based Routing

One of our routing streams was based upon a probabilistic classification system developed at Lockheed Martin. This system has been in development for less than one year, so many of its parameters have not yet been optimized. Also, only capitalized, stemmed single words are used as terms. Bi-grams, phrases, and extracted information will be added in the future.

The system generates routing scores for documents using three complementary components.

1 . A probabilistic scorer, which assigns a score to a document for a topic based upon the probability that the document belongs to the topic. Estimated probabilities for distinguishing terms (number of term occurrences divided by the number of occurrences of all terms in the training set) are gathered from training documents, and the union of all of the distinguishing terms for all of the topics defines a multinomial distribution. In considering a document to be routed, these estimated probabilities are combined with the count of terms which exist in the document to determine the probabilities that the document belongs to each of the topics [Guthrie et al, 1996].

The probabilistic scorer as described above has two features which need to be overcome to get good routing scores in a TREC type of situation. First, for each term there is a very good probability that the term does not occur in a document. This leads to good scores for documents which have none of the distinguishing terms for a topic. Second, even

for documents which have some of the distinguishing terms for a topic, no special weight is given to key terms in the query. This leads to good scores for 'near miss' documents, for example, documents about Russian joint ventures scoring highly for Topic 3, Japanese Joint Ventures.

2. The first feature, good scores for documents which contain no distinguishing terms, is overcome by including a score based upon the document frequency (number of relevant topic training documents which contain the term divided by the number of topic training documents) of 'required terms'. For each topic, the document is compared to a list of 10 to 20 of these terms, which is usually a subset of the distinguishing terms. If none of these terms for a topic occur in the document, the document is removed from consideration for that topic. For those required terms which do occur in the document, a score is calculated which is a sum of a function of the document frequencies of these terms. In addition to eliminating good scores for documents which contain no distinguishing terms, this score complements the probabilistic score because the probabilistic score does not increase the importance of terms which occur only once in most of the topic training documents, thus having a fairly low estimated probability but a high relevancy, or decrease the importance of terms which occur many times in only one of the topic training documents, thus having a fairly high estimated probability but a low relevancy.

3. The second feature, good scores for near miss documents, is overcome by including a score for each topic based upon a manually written boolean expression for each topic. If a document contains terms which fulfill the expression it gets a boost to its score, but if it does not fulfill the expression the document is not eliminated from consideration. The boolean expression helps weed out near misses, while not overly penalizing those documents which are relevant but which do not fulfill the expression.

For a new routing system the performance was satisfactory, scoring just below the median 11 point average precision. Unfortunately, the software had a small bug in the probabilistic portion which reduced the score somewhat. Due to a misplaced 'else' statement, a counter which was supposed to be counting the number of terms which were not distinguishing for any topic was actually counting, for each topic, the number of words which were not required terms. This resulted in a count which was incorrect, but in a rough sense was about 50 times what it was supposed to be, giving a final result which was still reasonable. Correcting the error increased the 11 point average precision over 45 topics from 0.1867 to 0.2299, a 23% improvement, and the number of relevant document retrieved from 2798 to 3375, a 21% improvement.

Future improvements for this system include the consideration of bi-grams, phrases, and extracted information, additional automation in term selection and boolean expression creation, improved stemming (the current stemmer is rule-based), and optimization of all of the parameters.

9.0 SUMMARY OF RESULTS

We submitted the total of 6 official runs in the main evaluation, and 4 official NLP track runs. In addition, 2 NLP baseline runs using SMART system have been submitted.

9.1 Adhoc runs

Adhoc runs were all in category A (entire 2 GByte database). The following 4 ad-hoc runs were submitted:

- **genrl1**: automatic run, short queries, with auto feedback on top 10
- **genrl2**: automatic run, long queries
- **genrl3**: manual run, long queries, with query expansion
- **genrl4**: manual run, long queries, with query expansion and auto feedback on top 10

An automatic run means that there was no human intervention in the process at any time. A manual run means that some human processing was done to the queries, and possibly multiple test runs were made to improve the queries. A short query is derived using only one section of a TREC-5 topic, namely the DESCRIPTION field. A long query is derived from any or all fields in the topic. An example TREC-5 query is show below; note that the Description field is

what one may reasonably expect to be an initial search query, while Narrative provides some further explanation of what relevant material may look like. The Topic field provides a single concept of interest to the searcher; it was not permitted in the short queries.

```

<top>
<num> Number: 252
<title> Topic: Combating Alien Smuggling
<desc> Description:
    What steps are being taken by governmental or even private entities world-wide to stop
    the smuggling of aliens.
<narr> Narrative:
    To be relevant, a document must describe an effort being made (other than routine border
    patrols) in any country of the world to prevent the illegal penetration of aliens across bor-
    ders.
</top>

```

TABLE 2. Precision changes across Ad-Hoc runs

PRECISION	GENRL1 automatic	GENRL2 automatic	GENRL3 manual	GENRL4 manual
11pt. average	0.1524	0.2093	0.2847	0.2741
%change		+37%	+87%	+80%
R-Precision	0.1965	0.2441	0.3126	0.3042
At 10 docs	0.3064	0.3809	0.5191	0.3042
At 100 docs	0.1694	0.3000	0.2615	0.2604

9.2 Routing runs

Routing submissions included two official runs:

- genrl5: automatic run, long queries, standard SMART+Prise run
- genrl6: automatic run, long queries, using multi-bin classification approach

A mistake was made in selecting weighting scheme in SMART portion of genrl5: a wrong weighting scheme (lnc) was accidentally used for indexing the pivotal stem stream in the test collection. Once we re-build the index with the correct lnc weighting, the result have improved substantially, as shown in the table below. The average precision checked against the summary posted by NIST shows now 4 queries at the best, 28 above median, and 13 below median. Note that this correction is independent of any specific database or queries.

TABLE 3. Average Precision on 45 routing queries: GENRL5

IR ENGINE	Corrected Prec.	Corrected R-Prec.	Official Prec.
SMART	0.2755	0.3145	0.0631
PRISE	0.2099	0.2473	0.2099
GENRL5	0.3023	0.3359	0.1968

The merge of Smart and Prise improved 9.7% on the average precision over the best individual component. The classification run (genrl6) was a merge of a classification scheme routing developed at Lockheed Martin, and standard

Prise routing used in previous TRECs. Again, a correction of a small error in the classification scheme improves our results as shown below.

TABLE 4. Average Precision on 45 routing queries: GENRL6

IR ENGINE	Corrected Prec.	Official Prec.
LM CLASS.	0.2299	0.1867
%change	+24%	
PRISE	0.2099	0.2099
GENRL6	0.2575	0.2222
%change	+16%	

Note that the classification scheme did quite well, outperforming (after correction) the standard Prise routing by some 10%. Our merging algorithm is also performing well consistently adding some 10% precision, provided that component runs are themselves relatively good.

10.0 NLP TRACK

The NLP Track was a specialized smaller-scale evaluation to experiment with more advanced NLP techniques. Our focus this year was on (1) evaluating value of special-purpose terms such as foreign-country references, and single-sense words, and (2) approaches to automatic full-text query expansion methods.

10.1 Automatic runs in NLP Track

We generated three automatic runs for the NLP track: **genlp1**, **genlp2**, **genlp3**. We focused on experimenting NLP related automatic query enhancement techniques. **genlp1** is the automatic run that uses the same multi-stream retrieval model as used in **genrl2** main evaluation run, with two added enhancements: foreign country tagging and hyphenated phrases tracking. In **genlp2**, we tested foreign country tagging exclusively against the corresponding SMART baseline (**sbase2**). Finally, in **genlp3**, we attempted to explore means of automating the full-text query expansion technique used in manual runs (**genrl3**, **genlp4**). In addition to the official runs, we also discuss an experiment with weighting of single-sense words in section 11.1.4.

10.1.1 Hyphenation

We used occurrences of hyphenated phrases in text as a guide for extracting other multi-word terms for indexing. Many semi-fixed expressions in English are occasionally hyphenated, which may indicate that their non-hyphenated occurrences should also be treated as single terms. In this experiment, we collected all the hyphenated words from the corpora (the less meaningful ones are eliminated by setting a threshold on the number of occurrences they appear), such as: *alien-smuggle*, *quick-freeze*, *roto-router*, *heart-surgery* *cigarette-smoke*, *lung-cancer*, *per-capita*, etc. The next step was to identify all the occurrences of those phrases in the collection and in the queries where they were not hyphenated and add the normalized term.

Unfortunately, the result of the stems stream shows slight deterioration comparing with performance before adding the hyphenated phrases. Among the 33 applicable queries, 22 queries show precision loss (on the average 43% per query) and only 11 queries show improvement (on the average 45% per query).

10.1.2 Foreign Country Tagging

For queries involving references to foreign countries, either direct, or indirect, e.g., good of foreign manufacture, we added special tokens for each reference of the concept 'foreign'. The identification is done simply by looking up certain key words and phrases (e.g. *foreign*, *other countries*, *international*, etc.). Using a list of foreign countries and

major cities acquired from the Internet, we tagged the documents in the collection with the same special “foreign” token whenever a foreign country or city was mentioned.

Only 10 queries (out of 45) were affected. Comparing with the base run, 9 out of these 10 show improvement, and only one shows a modest 5% performance loss. On the average, the precision gain is 27% for those queries. The result may suggest that type-tagging of terms in general, e.g., people, organizations, locations, etc. may lead to significant improvement in retrieval accuracy, a subject that has been the focus of much debate in Tipster community. The challenge is to identify a sufficient number of basic categories so that they can be found in a number of different queries, and such that an efficient object identifier can be implemented for them.

10.1.3 Concept Expansion

In our manual runs (**genrl3**, **genrl4**, **genlp4**), we tested the full-text query expansion, in which original queries were liberally augmented with text copied from database documents. The results were most encouraging, which prompted us to investigate ways of performing such expansions automatically.

One way to approximate the manual text selection process, we reasoned, is to focus on those text passages that refer to some key concepts identified in the query, for example, “alien smuggling” for query 252 and “cryonic suspension” for query 253.

The key concepts (for now limited to noun phrases) are identified by their repetitions as well as their relative locations within the query, e.g., in the title. We then take the top 100 retrieved documents for each query in an unexpanded query run, e.g., **genlp1**, and extract all paragraphs which contain references to any of the key concepts identified in the original query. These paragraphs are the pasted verbatim into the query. The original portion of the query may be saved in a special field to allow differential weighting. The expanded query were then run against the baseline index, producing **genlp3** ranking. Please note that, unlike **genrl3**, or **genlp4**, this run uses only one stream, namely stems stream, additionally augmented with SMART bigram phrases. This means that direct comparisons with any manual extension runs may not be valid here.

The above, clearly simplistic technique has produced some interesting results. Out of the fifty queries we tested, 34 has undergone the expansion. Among these 34 queries, we noted precision gains in 13, precision loss in 18 queries, with 3 more basically unchanged. However, for these queries where the improvement did occur it was very substantial indeed: the average gain was 754% in 11-pt precision, while the average loss (for the queries that lost precision) was only 140%. Overall, we still can see a 7% improvement on all 50 queries (vs. 40%+ when manual expansion is used).

In conclusion, the experiment shows that picking up the right paragraphs from documents to expand the query can indeed improve the performance dramatically. The future challenge is to devise a more precise automatic means to select those “good” paragraphs.

10.1.4 Single-sense Enhancement

Many words, when considered out of context, display more than one sense in which they can be used. When such word are used in actual text they may assume any of their possible senses, which can only be determined by examining the context. This has been a problem for word-based IR systems, and have spurred a number of largely unsuccessful attempts at sense disambiguation in text indexing. Another way to address this problem is to focus on words that do not have multiple-sense ambiguities, and treat these as special, because they seem just more reliable as content indicators.

We found that single-sense⁴ words tend to be more specific and thus more informative. We added more weight to terms that we considered low-number-of-senses, with single-sense words receiving the highest premium (duplicate standard $tf*idf$ weight). The results were mixed with the average precision gaining a modest 4.6%.

10.2 Manual run in NLP Track

genlp4 is a counterpart to the **genrl3** adhoc run. It is the only manual run we submitted in the NLP track, which uses the manually expanded queries against the multi-stream indexes. The goal is to see how the system performs with relatively high quality and long queries and compare it with the Smart baselines.

The evaluation over 45 queries of **genlp4** and the comparison with other manual NLP runs are shown in the table below. The average precision of most queries are above median.

TABLE 5. Manual NLP track run with query expansion (GENLP4)

Recall	814 out of 1064
11-pt Avg. Precision	0.3176
R-Precision	0.3090
Queries with best avg. precision	22
Queries with above avg. precision	17
Queries below average	6

To further comparing the retrieval performance of our multi-stream model and Smart with bi-grams, the un-processed manually expanded queries are then run against the Smart baseline index, the result shows noticeable improvement.

10.3 Summary of NLP Track runs

The following runs were obtained:

genlp1: automatic multi-stream run with foreign country tagging and hyphenated phrases.

genlp2: automatic single-stream run (stems and bigrams) with foreign country tagging.

genlp3: automatic, single-stream run with automatic the full-text query expansion.

genlp4: manual, multi-stream run with manual full-text query expansion.

sbase1: SMART baseline with stems and bigrams on "short" queries

sbase2: SMART baseline with stems and bigrams on "long" queries

sbase3: SMART baseline with stems and bigrams on full-text expanded queries

4. Or near-single-sense words, with a predominant single-sense. Generally, the fewer senses a word can have, the more reliable index term it appears to be.

TABLE 6. Precision improvement in NLP Track runs using

PRECISION	SBASE1	SBASE2	GENLP1	GENLP2	GENLP3	SBASE3	GENRL4
11pt. average	0.1478	0.2078	0.1773	0.2083	0.2220	0.2992	0.3176
%change		+41.0	+20.0	+41.0	+50.0	+102.0	+115.0
R-Precision	0.1609	0.2176	0.1776	0.2121	0.2242	0.3074	0.3091
%change		+35.0	+10.0	+32.0	+39.0	+91.0	+92.0
At 10 docs	0.1578	0.2044	0.2044	0.2044	0.2089	0.3089	0.3156
%change		+30.0	+30.0	+30.0	+32.0	+96.0	+100.0
At 100 docs	0.0544	0.0696	0.0664	0.0713	0.0709	0.0929	0.0998
%change		+28.0	+22.0	+31.0	+30.0	+71.0	+83.0

10.4 Stream Performance Evaluation

The weighting schemes used in genlp1 and genlp4 are the same:

TABLE 7. Stream weighting in NLP Track runs (genlp1 & 4)

STREAM	Weighting Scheme
Stems	Inc.ntn
Phrases	ltn.ntn
H+M Pairs	ltn.nsn
Names	ltn.ntn

Selecting the optimal weighting for each stream is essential. The issue is further complicated by the fact that the optimal weighting vary from collection to collection. The weighting schemes used in the submitted genlp1 and genlp4 appear reasonable, but they are not optimal. Had we chosen Inc.ltn weighting on the stems stream, genlp1 would have moved up to 0.1883 and genlp4 up to 0.2792.

The average precision over 45 queries stream-wise are:

TABLE 8. How different streams perform relative to one another (11-pt avg. Prec)

STREAM	genlp1	genlp4
Stems	0.1682	0.2626
Phrase	0.1233	0.2365
H+M Pairs	0.0755	0.2040
Names	0.0844	0.0608

The average precision of different stream merging with the strongest single-stream(stems) retrieval are shown in Table 9.⁵

TABLE 9. How merging improves precision, wrt. queries used

Which Streams MERGED?	genlp1 %change	genlp4 %change
all 4	+5.4	+20.94
Stems+Phrases+Pairs	+6.6	+22.85
Stems+Phrases	+7.0	+24.94
Stems+Pairs	+2.2	+15.27
Stems+Names	+0.6	+ 2.59

The results indicate that syntactic phrases seem to be more effective with longer the queries.

10.5 Baseline Runs

Two baselines were generated for the NLP track: **sbase1** using “short” queries (the <desc> field only) and **sbase2** that utilizes all the fields in the adhoc topics (“long” queries). The document test collection was category B (250MBytes Wall Street Journal data). Both baselines were obtained using standard SMART processes, including “statistical phrase” terms, i.e., high frequency adjacent word pairs (bi-grams).⁶

11.0 Experiments in Stylistic Analysis

Texts vary not only by topic. *Stylistic* variation between texts of the same topic is often at least as noticeable as the *topical* variation between texts of different topic but same genre or variety; style is, broadly defined, the difference between two ways of saying the same thing.

Stylistic variation in a given text can occur in many ways and on many linguistic levels: lexical choice, choice of syntactic structures, choice of cohesion markers on a textual level, and so forth. In these experiments we measure several different types of simple *stylistic items*: lexical statistics such as average word length, long word counts, type/token ratios, pronoun counts and digit counts; syntactic statistics such as average sentence length and some parsing statistics and combine them using multivariate techniques (Karlgrén and Cutting, 1994). We use the Wall Street Journal corpus for TREC-4 as a training set: we have attempted to find statistically significant stylistic differences between documents that have been judged relevant for some query on the one hand and documents that were not judged relevant for any query at all on the other.

We did find such differences; for most metrics tested, the difference was statistically significant even by univariate⁷ tests, even if the difference between texts retrieved by some system and non-retrieved texts was larger by far than the difference between relevant and non-relevant. In summary, our results are that retrieved highly ranked texts - both relevant and non-relevant - are longer⁸, with a more complex sentence structure than the rest of the corpus, and that rel-

5. First, all four stream participate the merge (officially submitted). Second, the name stream is taken away; Next, the pair and name stream are taken away; Next, the phrase and name stream are left out. Finally, the phrase and pair stream are left out.

6. We had to resubmit the baselines after the official results were obtained from NIST, because of a mistake in the term weighting scheme used.

7. Mann Whitney U

8. Which also has been observed, pointed out, and utilized by the Cornell research group at the latest TREC conference (Buckley et al, 1995).

evant texts differ from nonrelevant in that they tend towards more complexity - textual, syntactic, and lexical - on most measurements. Moreover, we found that these differences varied for subsets of the Wall Street Journal: certain types of article had a higher percentage relevant documents than others. (Karlgrén, 1996).

11.1 Visualizing Stylistic Variation

Stylistic variation is not unrelated to topic: obviously certain topics will be more formal than others; others will be more technical; yet others more discursive. How much user preferences for this are reflected in the query itself or the initially retrieved set of documents is an open question. We envision using stylistic data primarily in an interactive setting, in a display tool as a user-manipulable filter for interactive retrieval.

We have a prototype tool which will compute stylistic statistics for a set of texts, and which will then display the texts in a 2-D plot for any pair of statistics chosen. It is clear that using simple variables in this way is risky: firstly, what variation the display models is unclear to the user; secondly, risk of random or chaotic variation is great. We have experimented with combining scores from the various variables in linear weightings using principal components analysis (Karlgrén and Straszheim, 1997).

A useful strategy might be to pick a couple of parameters with a seemingly high spread. As example material we use here the top 50 retrieved texts for our system for query 203 "What is the economic impact of recycling tires?" together with the top 50 texts retrieved by Altavista from the World Wide Web for the same query. We find an example pair of variables which seems to disperse the material quite well as shown in Figure 12.1. The WWW material is marked with open circles; the TREC data with filled squares. Unsurprisingly, the WWW material is stylistically more heterogeneous than the TREC data: the TREC outliers are Patent and Federal Register texts.

11.2 Stylistics As A Way To Improve Precision

Now, the realization that stylistic variation is related to topical variation led us to perform some experiments specifically to improve our average precision in the TREC evaluation. We took some queries from previous years and used a system essentially like the one we used for this year's submissions to produce rankings for the documents.

We then tried to find methods that would identify non-relevant documents from the list of 1000 retrieved and submitted documents. These documents would then be moved to the end of the list, hopefully improving the evaluation results for the query. We used the C4.5 classification tool (Quinlan, 1993) which takes multivariate material and produces simple rules to partition items into classes using the variable values.

We knew from our first experiments that there are statistically significant differences between relevant and non-relevant documents in the TREC Wall Street Journal material as a whole. So far, so good, but the problem comes trying to apply these results on a query-by-query basis. The sets retrieved for each query are different stylistically; the genres and styles vary from topic to topic and thus from query to query. If we learn rules to distinguish relevant from non-relevant for the entire corpus and try to apply the rules across queries, we find we either degrade performance or at the least do not improve it.

The two rules below are examples of this. We found that these two rules work quite well for their respective training corpora and when tested on this year's material they improve results for many queries as can be seen in Figure 12.2. However, both rules suffer breakdowns on at least one query; this reduces the advantage gained from the other queries so that average precision is a tad lower than for the unordered set.

The consequence is that to make use of stylistic variation for reliable relevance grading we need a query typology: each query must be identified for likely style preferences. This remains future work and results from experiments in query categorization are pending.

Rule 4: *Digits* > 0.0179775 -> class non-relevant [95.0%]

Rule 876: *Word count* <= 1308 &
 Chars per word <= 5.10619 &
 Digits > 0.0119782 &
 Words per sentence > 17.4583 &
 1st person pronouns <= 0.0166667 &
 Persuasive adverbs <= 0.00234467 -> class non-relevant [90.3%]

TABLE 10. Effect on Ave. Precision

Baseline PRISE	Using Rule 4	Using Rule 876
0.1460	0.1459	0.1452

12.0 CONCLUSIONS

We presented in some detail our natural language information retrieval system consisting of an advanced NLP module and a 'pure' statistical core engine. While many problems remain to be resolved, including the question of adequacy of term-based representation of document content, we attempted to demonstrate that the architecture described here is nonetheless viable. In particular, we demonstrated that natural language processing can now be done on a fairly large scale and that its speed and robustness has improved to the point where it can be applied to real IR problems.

The main observation to make is that natural language processing is not as effective as we once hoped in to obtain better indexing and better term representations of queries. Using linguistic terms, such as phrases, head-modifier pairs, names, or even simple concepts does help to improve retrieval precision, but the gains remained quite modest. On the other hand, full text query expansion works remarkably well. Our main effort in the immediate future will be to explore ways to achieve at least partial automation of this process. An initial experiment in this direction has been performed as part of NLP Track (genlp3 run), and the results are encouraging.

ACKNOWLEDGEMENTS. We would like to thank Donna Harman of NIST for making her PRISE system available to us since the beginning of TREC. Will Rogers provided valuable assistance in installing updated versions of PRISE at NYU and Rutgers. We would also like to thank Ralph Weischedel for providing and assisting in the use of the BBN's part of speech tagger at NYU. This paper is based upon work supported in part by the Advanced Research Projects Agency under Tipster Phase-2 Contract 94-FI57900-000, and the National Science Foundation under Grant IRI-93-02615.

REFERENCES

- Chris Buckley, Amit Singhal, Mandar Mitra, Gerard Salton. 1995. "New Retrieval Approches Using SMART: TREC 4". In Proceedings of TREC4.
- Guthrie, Louise and Leistensnider, James, 'A Simple Probabilistic Approach to Classification and Routing', Proceedings of the TIPSTER Text Program Phase II Workshop, Sponsored by Defense Advanced Research Projects Agency, May 6-8, 1996.
- Jussi Karlgren. 1996. "Stylistic Variation in an Information Retrieval Experiment" In Proceedings NeMLaP 2, Bilkent, September 1996. Ankara: Bilkent University. (In the Computation and Language E-Print Archive: cmp-lg/9608003).
- Jussi Karlgren and Douglass Cutting. 1994. "Recognizing Text Genres with Simple Metrics Using Discriminant Analysis", Proceedings of COLING 94, Kyoto. (In the Computation and Language E-Print Archive: cmp-lg/9410008).

Jussi Karlgren and Troy Straszheim. 1997. "Visualizing Stylistic Variation." In the Proceedings of the 30th HICSS, Maui.

J. Ross Quinlan. 1993. C4.5: Programs for Machine Learning. San Mateo: Morgan Kaufmann.

Strzalkowski, Tomek and Jose Perez-Carballo. 1994. "Recent Developments in Natural Language Text Retrieval." Proceedings of the Second Text REtrieval Conference (TREC-2), NIST Special Publication 500-215, pp. 123-136.

Strzalkowski, Tomek, Jose Perez-Carballo and Mihnea Marinescu. 1995. "Natural Language Information Retrieval: TREC-3 Report." Proceedings of the Third Text REtrieval Conference (TREC-3), NIST Special Publication 500-225, pp. 39-53.

Strzalkowski, Tomek, Jose Perez-Carballo and Mihnea Marinescu. 1996. "Natural Language Information Retrieval: TREC-4 Report." Proceedings of the Third Text REtrieval Conference (TREC-4), NIST Special Publication 500-2xx.

Strzalkowski, Tomek. 1995. "Natural Language Information Retrieval" Information Processing and Management, Vol. 31, No. 3, pp. 397-417. Pergamon/Elsevier.

Strzalkowski, Tomek, and Peter Scheyen. 1993. "An Evaluation of TTP Parser: a preliminary report." In H. Bunt, M. Tomita (eds), "Recent Advances in Parsing Technology." Kluwer Academic Publishers, pp. 201-220.



CLARIT Compound Queries and Constraint-Controlled Feedback in TREC-5 Ad-Hoc Experiments

Natasa Milic-Frayling¹, Xiang Tong², Chengxiang Zhai², David A. Evans¹

¹CLARITECH Corporation
5301 Fifth Avenue
Pittsburgh, Pennsylvania 15232-2124

²Laboratory for Computational Linguistics
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3890

1. Introduction

A fundamental problem for searching over large databases in ad-hoc mode is *the formulation of an effective initial query* that is both comprehensive and focused. The query needs to be comprehensive enough to retrieve, on its own or enhanced by various automatic feedback techniques, relevant documents that possibly address different aspects of the topic. At the same time, it has to be focused enough to ensure quality input into intermediate feedback processes or high precision in the final retrieval.

In TREC, the initial query formulation problem has been addressed in two types of ad-hoc tasks: the *automatic ad-hoc task* that essentially relies on the original description of the TREC topic, a complete topic or some part of it, and the *manual ad-hoc task* in which users create queries either solely based on their own understanding and knowledge about the topic or by consulting various information resources, including documents from the target database, as permitted in this year's TREC.

We are particularly interested in the manual ad-hoc task as it highly resembles real world search situations, especially by allowing the user to use information from the target corpus. Two most natural ways to exploit the target databases directly are (1) to allow the user to re-formulate the query through interactive searches and (2) to enhance the initial query automatically based on documents reviewed by the user. Some of the manual ad-hoc experiments in TREC 5 include both methods, others use the latter.

The main objective of our TREC-5 ad-hoc experiments is to evaluate a method by which the user can *influence* rather than perform the selection of feedback documents for an automatic query enhancement, assuming that the active interaction between the user and the system ends with the initial interactive search over the target data. In this respect CLARITTM¹ TREC-5ad-hoc experiments represent a continuation and further refinement of the study on *constraints controlled feedback* that we initiated in TREC-4 ad-hoc experiments. However, this year we performed ad-hoc experiments with the CLARIT commercial system, which, in contrast to the CLARIT experimental suite, does not yet incorporate system features that have been effective in improving retrieval performance in previous TRECs, such as partitioning documents into fixed length overlapping windows and using automatic negative feedback in the form of the CLARIT distractor term space [1, 2]. The CLARIT commercial system, on the other hand, is equipped with a GUI that fully supports the user in interactive generation of CLARIT *compound queries*, a natural language (NL) query supplemented by Boolean type constraints, that we previously explored in a more rudimentary form in TREC 4 [2].

The second objective of our study is to determine how effective user specified constraints are in facilitating the final ranking of documents in order to achieve a higher front-end precision. Our official submissions, CLTHES and CLCLUS, explore the use of constraints to control both the automatic feedback and the final ranking of documents.

Finally, we include in our analysis a new experimental feature of the CLARIT system, namely manual query expansion using pre-computed concept clusters from the target database.

Generally there are many difficulties associated with experimental designs that rely upon an authoritative document relevance judgment that is independent of the experiment process. In manual ad-hoc experiments this problem is particularly emphasized since the characteristics of individual searchers become more pronounced and

¹CLARIT is a registered trademark of CLARITECH Corporation.

influential with increased levels of user interaction with the system. In relation to our TREC-5 experiments we anticipate difficulties in differentiating the effects of:

- the discrepancy between the user's and a NIST expert's relevance criteria
- the ability of the users to translate their relevance criteria into an 'operational definition' expressed through Boolean type constraints
- the inherent limitation of constraints in characterizing document relevance.

We are also sensitive to the problem of *over-fitting of the initial query*, in particular the set of constraints, to the limited number of documents that the user reviews during the process of creating the initial query. In contrast to the relevance judgment problem that is inherent to our experiment design, user bias introduced through the query construction process is a problem that needs to be dealt with in real life applications. It exists in all situations in which the user has a limited view of the information space. Thus, an important issue that will be a focus of our future experiments is the representation of the global information space as an aid to document retrieval. Although still in an experimental form, the concept clustering technique used in CLCLUS experiment represents our first step in that direction.

In Section 2 of this paper we present a detailed description and analysis of the experiments with feedback control. In Section 3 we discuss the official TREC-5 experiments, CLTHES and CLCLUS. We summarize our findings in Section 4. The Appendix contains information about system parameters and experiments performed in the TREC-5 Very Large Collection (VLC) track.

2. Experiments with Feedback Control

2.1 CLARIT Compound Query

The general CLARIT approach to TREC retrieval tasks relies on a rich representation of both TREC topics and documents in TREC databases. CLARIT TREC queries are often composed of several layers of terminology that predominantly originate from the target data [2]. It is essentially the overlap between the statistically prominent features in the query term space and the document terms space that determines the degree of document relevance to the topic. This retrieval strategy generally reduces the ambiguity of query concepts by providing adequate contextual description and consequently improves retrieval precision and recall. However, the key to the procedure is a relatively difficult task of identifying good sources of terminology in the target corpus and effective methods for extracting such terminology.

Normally, in ad-hoc experiments the top N documents or document windows retrieved in response to the initial query are assumed to be relevant to the query and further processed by the CLARIT Thesaurus Extraction module to obtain the terminology characteristic of the selected documents [1, 2]. This terminology, when added to the initial query, helps the system identify documents with similar content.

Although the thesaurus extraction technique is robust and provides effective query augmentation even when the precision of the initial retrieval is not very high, we wish to design a more reliable procedure for selecting potentially relevant documents for ad-hoc searching. For that purpose we create a notion of the *CLARIT compound query*, which consists of a natural language (NL) query and a set of Boolean type constraints constructed by the user. The constraints are intended to capture and enforce the user's relevance criteria during selection of documents for automatic feedback in a partially interactive or non-interactive search environment where the user has limited or no access to the target database. In such situations constraints can serve the general purpose of providing the user's input to intermediate automatic processes, such as automatic feedback, or propagating the user's relevance criteria even further through the retrieval process and facilitating the final ranking of documents.

Therefore, in our experiments that address feedback control, we allow the user to specify a compound query which, in addition to the NL query, contains the user's criteria for selecting documents expressed in the form of Boolean constraints. These constraints are applied as filters over documents retrieved in response to the NL query (see Figure 1). More precisely, the top N retrieved document windows are evaluated with respect to the user's selection criteria. Only those that satisfy the user's criteria are used for automatic feedback.

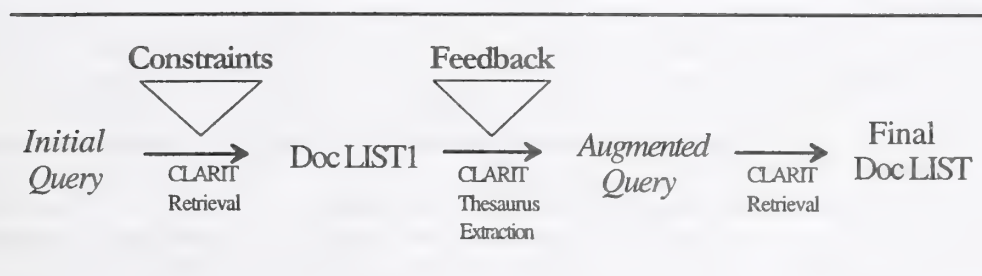


Figure 1: Document filtering as a mechanism to control automatic feedback

This approach to achieving more effective automatic feedback was first evaluated in our TREC-4 experiments in the context of different levels of control that the user may have over the feedback procedure [2]. In fact, in TREC 4 we committed ourselves to completing a sequence of experiments that explore the feedback control in more depth. In the subsequent sections we present the results that we obtained by performing these experiments on TREC-5 data.

2.2 Feedback Control Experiment Design

In our feedback control experiments we explore two factors:

1. *The type of the initial query*, i.e., the source of terminology used by the user to formulate the initial query; and
2. *The feedback control*, i.e., the level of control that the user has over the system's expansion of the query.

Initial Query Types	Feedback Control		
	Fully Automatic	Document Filtering	Manual Selection of Documents
NLP Query	○ A1	○ A2	○ A3
Terminology from Non-Target Databases	○ M1	○ M2	○ M3
Terminology from the Target Database	○ I1	○ I2	○ I3

Table1: Set of CLARIT Feedback Control Experiments

Based on the source of terms for the query, we classify queries into three categories:

1. *Queries created by automatic NL processing of the topic description*

Terms in such queries are derived directly from CLARIT NLP of the topic description or an information request. In fact, in our TREC-5 exploration of the feedback control we used queries that consist of a subset of the NLP-generated terms. This subset of terms was identified by the user to avoid terminology that is not directly related to the topic but serves only as descriptors of the retrieval task.

2. *Manually created queries supplemented with terms from a non-target corpus*

The user creates the query using the CLARIT IR system (in particular, thesaurus-discovery operations) to find useful terms from information sources other than the target corpus.

3. *Manually created queries supplemented with terms from the target corpus*

The user creates the queries using the CLARIT IR system over the target corpus.

The second parameter in the experiments represents different levels of control that the user has over the feedback procedure²:

1. Fully automatic query augmentation

The top N document windows retrieved by the initial NL query are used by the system to extract the CLARIT Thesaurus. A specified portion of thesaurus terms is automatically added to the initial query.

2. Filtering of feedback documents based on user specified constraints

The user specifies Boolean type constraints to be used to filter document windows retrieved by the initial NL query. Only the top N document windows that meet the user's constraints are used in thesaurus extraction. The specification of constraints may or may not include information about the target corpus.

3. User selection of feedback documents

The user is allowed to review documents that are retrieved in response to the initial NL query and select the ones to be used for automatic query augmentation.

2.3 Feedback Control Experiments with TREC-5 data

As mentioned before, we used document filtering to facilitate automatic feedback in TREC-4 ad-hoc experiments for the first time [2]. The results of those experiments showed a modest improvement of the retrieval performance. Our objective here is to examine the effects of similar feedback control using TREC-5 data and topics. We extend the analysis to all three types of manual queries described in Section 2.2 since they represent interesting classes of IR problems and applications. Indeed they represent an abstraction of typical situations such as searching with minimal user intervention, building of user's interest profiles without having example relevant documents for individual topics, or enhancing the basic interactive search by automatic query augmentation.

The main differences between the TREC-4 and TREC-5 experiments are in the implementation of the document filtering mechanism and the choice of the system used in the experiments:

(1) TREC-5 constraints are designed dynamically, using interactive searches over the non-target or target data, rather than being specified separately from the NL query.

(2) In addition to the logical AND and OR operators, the TREC-5 constraints use the NOT³ operator.

(3) In TREC 5 both the manual construction and the batch processing of queries are performed using the commercial CLARIT IR System rather than the CLARIT experimental system. This implies that instead of partitioning and matching documents on the level of fixed length overlapping windows we used fixed length disjoint document windows. Furthermore, we did not apply the CLARIT negative feedback technique to contrast the potentially useful terminology and the general, distracting terminology [2]. We also indexed individual TREC databases separately and used the CLARIT database merging technique to obtain the final rank list of documents, rather than creating one monolithic TREC-5 database.

The results of the TREC-5 feedback control experiments are summarized in Tables 2-4. For clarification, we give a brief description of each group of experiments, as follows:

Experiments with simple NL queries. Initial queries for A1-A3 experiments were created by reviewing the output of the NL processing of the topic text and eliminating frequent and non-specific terms. In the document filtering experiment (A2) we applied the same set of constraints that was generated through interactive searches over the target database and used in both the experiment I2 and the official run CLTHES. For completeness and consistency we will, in the future, test the NL queries with constraints generated only based on the topic description.

²Originally we considered an additional type of user feedback: selection of feedback documents and terminology to be added to the query. However, the simulation of such feedback by a NIST expert cannot be reliably implemented since we cannot reliably measure the degree to which our term selection approximates that of a NIST expert.

³In fact, the NOT operator was used in very few topics. Thus, in that respect TREC-4 and TREC-5 filtering mechanisms are comparable.

Experiments with queries constructed using terminology from non-target data. Initial queries for this set of experiments (M1-M3) were created through interactive searches over the AP89 database, which is not included in the TREC5 data set. Constraints used in M2 were constructed either dynamically, with verification of the effects that they have on the search over the AP89 data, or simply from the user's knowledge of the topic domain.

Experiments with queries constructed using terminology from the target data.

For these experiments (I1-I3) we formulated the compound query, i.e., both the initial NL queries and the associated constraints, by searching over the target data. The same NL queries and constraints are used in the official CLTHES run (see Section 3).

All the experiments with feedback based on the user selection of documents, A3-M3-I3, use the NIST relevance judgment of documents to simulate search by an expert user. Additional query terminology is extracted from the relevant documents that appear among the top 100 documents retrieved in response to the initial NL query. In the future we plan to re-run these experiments with feedback based on document windows rather than full documents, to make them more consistent and comparable with the rest of the discussed experiments. We, in fact, expect a higher recall and precision since, in our past experiments, thesaurus based query expansion using terminology from system discovered relevant portions of documents, rather than the text of complete documents, has proven to be more effective.

2.4 Experiment Results

The results of the three sets of experiments presented in Tables 2-4 and Figures 2-4 lead us to the conclusion that the controlled feedback technique, facilitated by user specified constraints, helps bridge the performance gap between automatic feedback (A1-M1-I1) in which the system extracts additional terminology from the top N retrieved documents windows and automatic query augmentation with terminology from the *truly relevant* documents (A3-M3-I3).

Indeed, filtering generally improves precision: an increase in average precision and R-precision is observed for all three types of queries. We suspect that the reason for a consistently higher precision in experiments with the compound query created interactively over the target data (I1-I2-I3) was achieved because of the high precision of the initial query and the nature of the thesaurus based query enhancement used in the feedback phase.

Although an initial query, such as the one created interactively over AP89 database (and used in the experiments M1-M2-M3), might provide a better general representation of the topic, as it could, perhaps, be inferred from the associated recall statistics, its initial precision over the target database may not be that high. Since the CLARIT Thesaurus extraction technique discovers terminology prominent in a given set of documents, its use in the feedback phase further emphasizes features of the initially retrieved documents. In that manner, higher initial precision yields a higher overall retrieval performance.

Initial Query Type	Feedback Control		
	Fully Automatic	Document Filtering	User Selection of Documents
NL Query	● A1	● A2	● A3
Recall (Max = 5,524)	2,859	2,778 (-3%) ¹	3,070 (7%) ¹ (11%) ²
Average Precision	0.1794	0.2168 (21%)	0.2853 (59%) (32%)
R-Precision	0.2213	0.254 (15%)	0.2913 (32%) (15%)
Front-end Precision	0.5005	0.5428 (8%)	0.8295 (66%) (53%)

¹Relative difference with respect to the fully automatic feedback exp. ²Relative difference with respect to the document filtering exp.

Table 2: Feedback control experiments with simple NL manual queries

Furthermore, although we expect that user constraints may inhibit recall, this is observed only in the experiments with the simple NL query combined with CLTHES constraints (A2). Similarly, front-end precision is higher for all searches with constraints except of the run M2.

Initial Query Type	Feedback Control		
	Fully Automatic	Document Filtering	User Selection of Documents
Terminology from Non-Target Databases	● M1	● M2	● M1
Recall (Max = 5,524)	3,279	3,322 (1%) ¹	3,386 (3%) ¹ (2%) ²
Average Precision	0.1974	0.2213 (12%)	0.3139 (59%) (42%)
R-Precision	0.2407	0.2616 (9%)	0.3184 (32%) (22%)
Front-end Precision	0.579	0.5562 (-4%)	0.8964 (55%) (61%)

¹Relative difference with respect to the fully automatic feedback exp. ²Relative difference with respect to the document filtering exp.

Table 3: Feedback control experiments with queries created using terminology from non-target databases

Initial Query Type	Feedback Control		
	Fully Automatic	Document Filtering	User Selection of Documents
Terminology from Target Databases	● I1	● I2	● I3
Recall (Max = 5,524)	3,116	3,144 (1%) ¹	3,210 (3%) ¹ (2%) ²
Average Precision	0.2261	0.2542 (12%)	0.3052 (35%) (20%)
R-Precision	0.2548	0.2933 (15%)	0.3195 (25%) (9%)
Front-end Precision	0.6703	0.7367 (10%)	0.8954 (34%) (22%)

¹Relative difference with respect to the fully automatic feedback exp. ²Relative difference with respect to the document filtering exp.

Table 4: Feedback control experiments with CLTHES manual queries

From the above experiments we can also make interesting observations regarding two important issues related to the manually created compound queries:

Over-fitting of the NL query

We suspect that the lower recall in the experiment I1 (with the initial NL query generated over the target data), in comparison to the recall achieved in M1 (the experiment with the NL query constructed over a non-target database) is a consequence of NL query over-fitting to the documents reviewed by the user during NL query construction. Having tried several search strategies and reviewed retrieved documents, the user probably focused on the aspects of the topics represented in the documents. Consequently, the created queries reflect the user's view and understanding of the topic based on a limited number of documents viewed from the target database.

Over-fitting of the user constraints

Experiments A1-A1-A3 are very useful for assessing the degree to which user constraints, independently from the NL query, are influenced by the user interpretation of the topic. It is interesting to note that, when combined with the constraints generated through interactive search over target database (I2), the automatically generated NL queries yield a slightly lower recall than with fully automatic feedback (A1). This is an indicator of over-fitting of manually built constraints to the documents reviewed by the user during manual building of queries.

Indeed, it seems that some of the features in the automatic NL query that were responsible for retrieving certain types of relevant documents were suppressed by the use of constraints. However, the decrease in recall in A2 was not followed by a decrease in retrieval precision. The reason for that is the robustness of the thesaurus based feedback. Loss of a relatively small percentage of relevant documents (3%) does not have a great impact on the types of terms extracted by the thesaurus technique. In fact, it seems that the user's constraints were successful in retaining a sufficient number of relevant documents and that the thesaurus extracted terms further amplified the role of query features responsible for retrieving relevant documents. This resulted in increased retrieval precision.

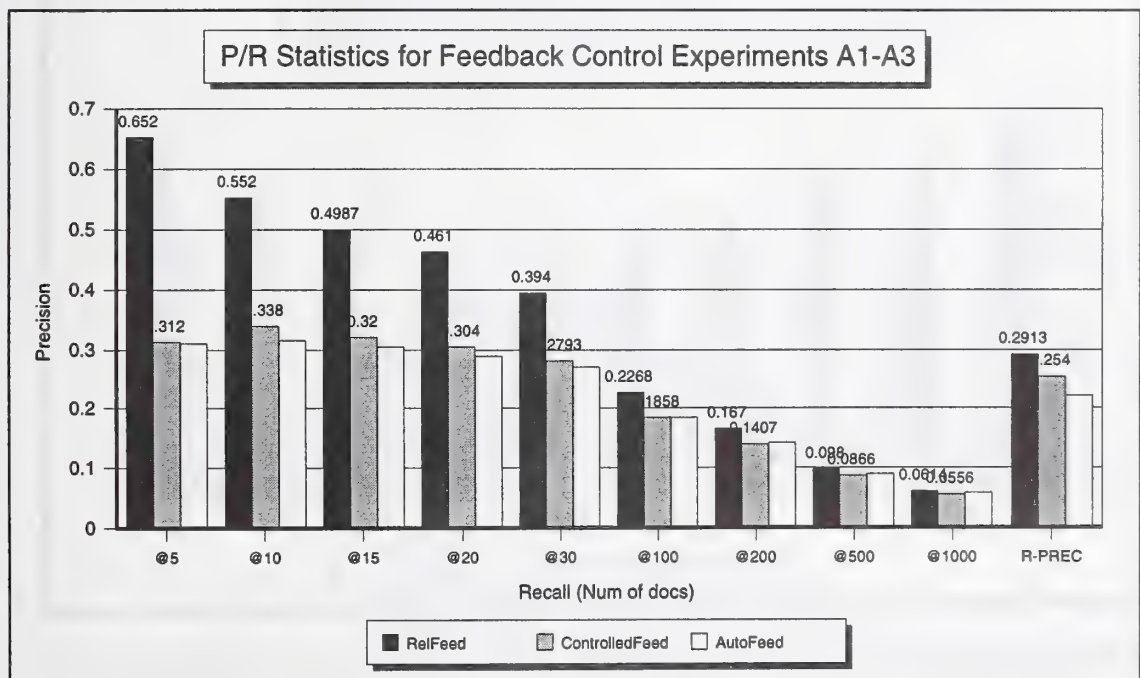
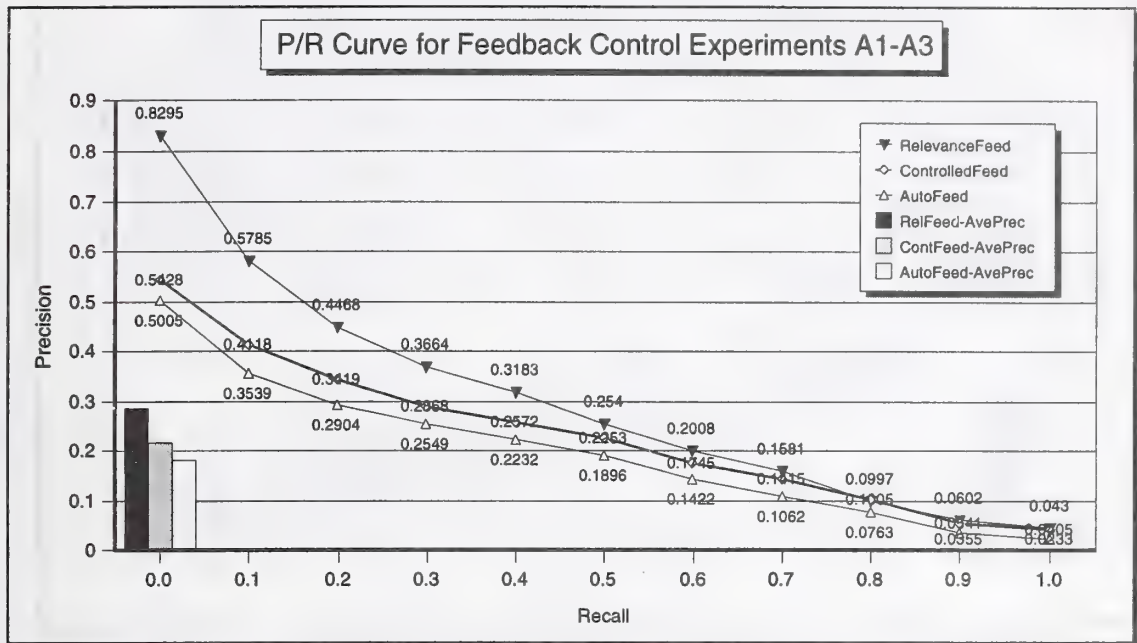


Figure 2: Precision/recall curves and statistics for the TREC-5 feedback control experiments A1-A3

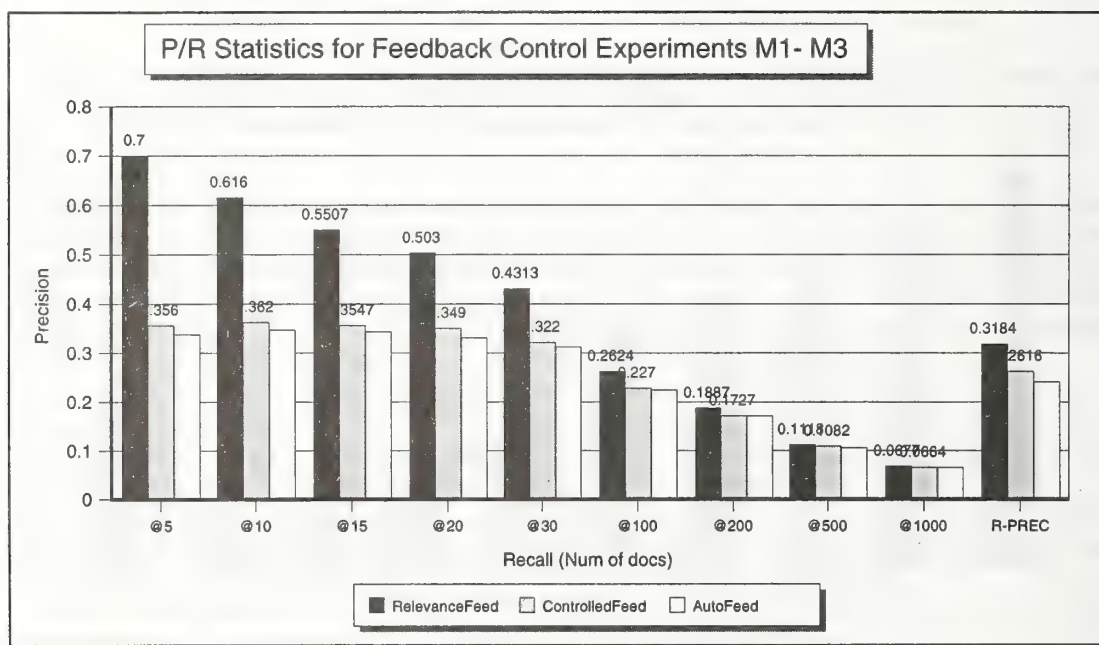
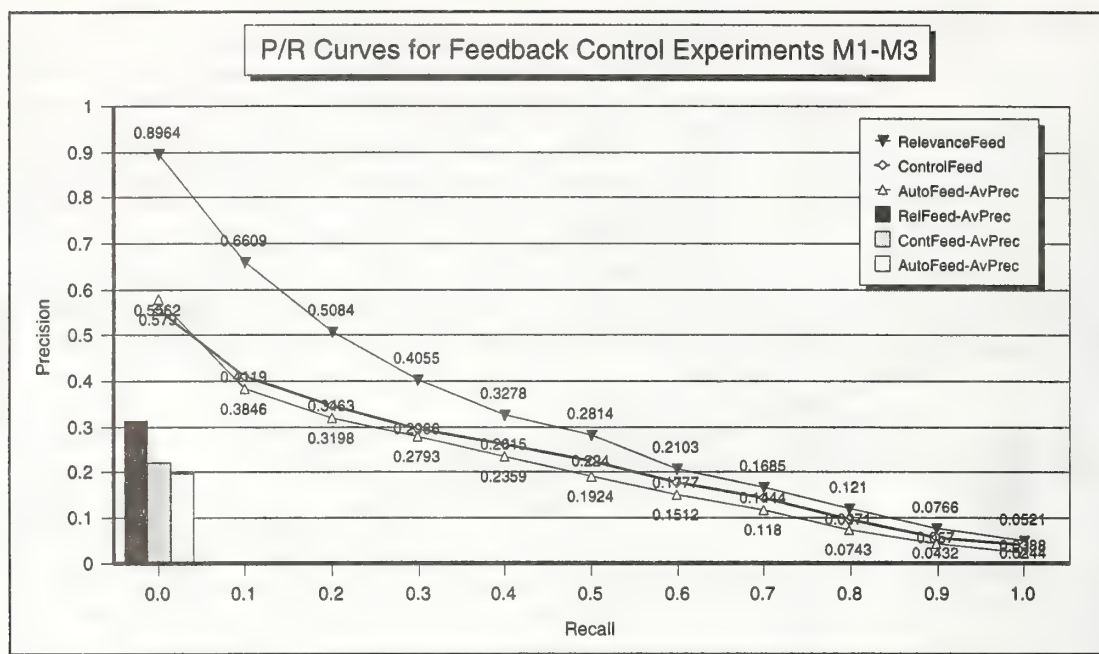


Figure 3: Precision/recall curves and statistics for the TREC-5 feedback control experiments M1-M3

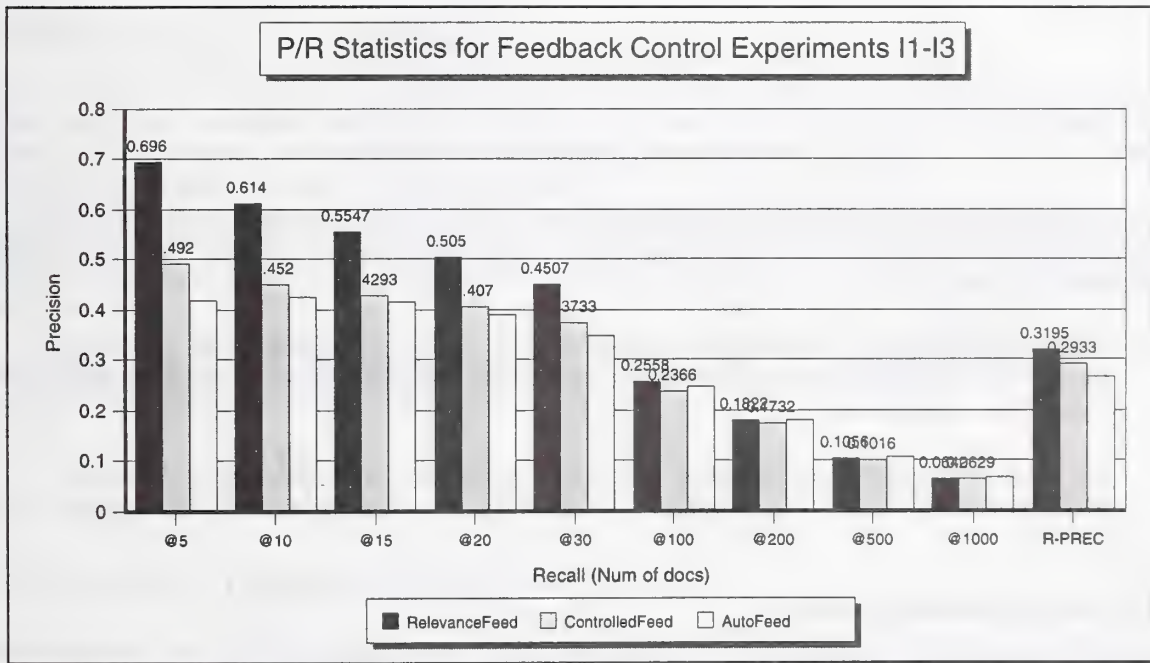
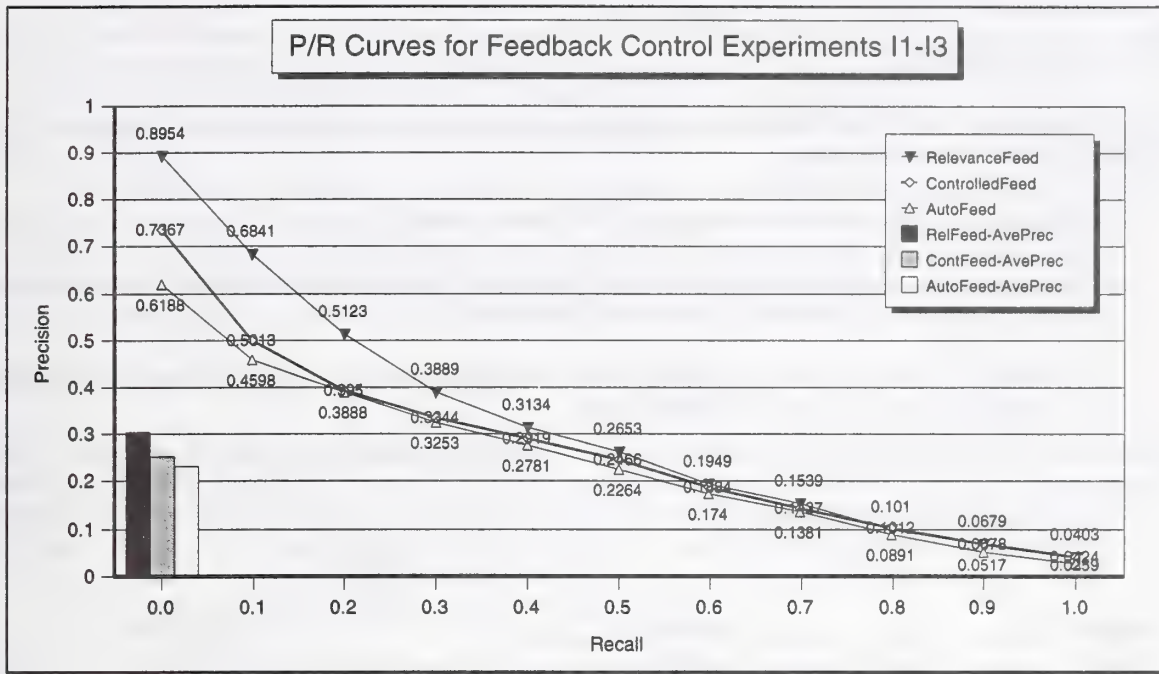


Figure 4: Precision/recall curves and statistics for the TREC-5 feedback control experiments I1-I3

3. CLARIT TREC5 Experiments: CLTHES and CLCLUS

3.1 Experiment Design

For the official submission of TREC-5 ad-hoc experiments we selected two experimental runs, CLTHES and CLCLUS, that include new techniques:

- (1) Creating the initial manual queries using CLARIT term clusters
- (2) Applying a second set of constraints to filter the results of the augmented query with the goal of improving the front-end precision of the retrieved documents
- (3) Merging the results from the constrained and unconstrained searches to obtain the final set of documents with a higher front-end precision and higher recall.

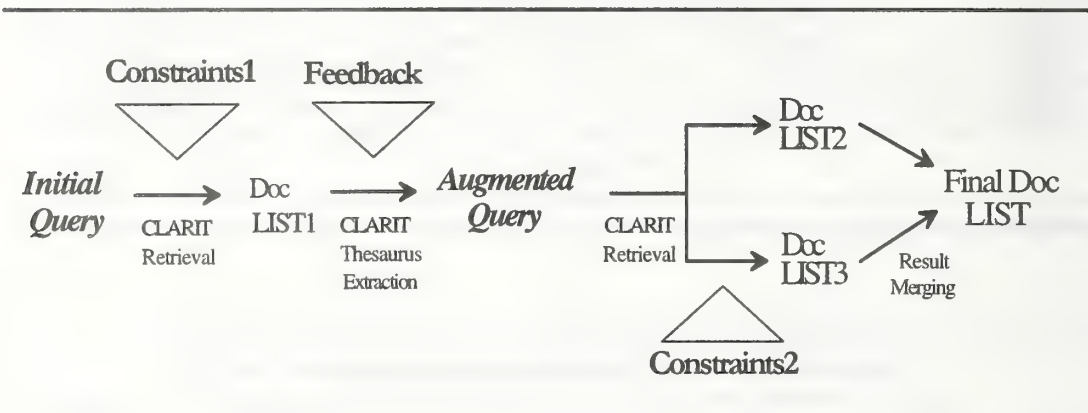


Figure 5: Design of the CLTHES and CLCLUS Experiments

Focusing more on exploiting constraints to improve the precision of the final retrieval and related result merging techniques, CLTHES and CLCLUS represent an extension of the feedback control experiments presented in Section 2. Their main objectives are:

- (1) To verify whether user input in the selection of documents for final retrieval, facilitated through a second set of constraints, would be beneficial for achieving better retrieval performance, in particular higher front-end precision
- (2) To explore result merging techniques that combine the benefit of the higher front-end precision observed with more tightly controlled searches with the higher recall expected from less constrained searches.

3.1.1. Construction of Queries

In CLTHES and CLCLUS experiments, CLARIT manual queries were created through iterative searching and review of documents from the target databases, TREC Disk 2 and Disk 4, using the CLARIT Interactive system.

The two experiments differ mainly in the source of terminology that the system provides to the user as an aid for creating the queries. In CLTHES the terminology is generated by the CLARIT Thesaurus extraction technique from user specified documents. More precisely, the user starts building a NL query by initiating a search with key concepts from the topic text and extracting CLARIT Thesaurus from the documents that the user finds relevant to the query. The user then reviews the thesaurus terms and selects those suitable for the query and/or the constraints. The NL queries and the Boolean constraints are created and tested simultaneously.

In the CLCLUS experiment we create for each TREC-5 database a complementary database of terminology clusters. Individual clusters consist of terminology that tends to co-occur in content related documents and essentially provide an overview of themes in the corpus. The clusters themselves are treated as documents by the CLARIT system and therefore can be explored using CLARIT search. In particular, the user can search over terminology clusters to identify those that best correspond to a given term or a set of terms. In CLCLUS the user selected the terms from individual clusters that seem most appropriate for describing the topic and added them to the query.

We view CLARIT terminology clustering as an alternative to the thesaurus based technique for constructing and automatically enhancing queries. The main advantage of using clusters to create a query is that the procedure does not involve document review and relevance assessment since the user is presented with already digested information. Furthermore, terminology clusters have a higher potential for capturing various aspects of a particular topic in the database than a thesaurus created from the top N initially retrieved or user specified documents.

3.1.2 Constraints and Merging of Results

The CLARIT System supports searching on the content of various fields in the documents either by NL querying alone or by supplementing the NL query with Boolean type constraints. In CLTHES and CLCLUS experiments we indexed the documents as having only two fields, the body of the text and the document title. The constraints formulated by the user were restricted to the body of the text. For example, for the Topic 279 we have:

Topic 279: 'Earth magnetic pole shifting'

Constraints 1: (DocHasTerm "earth") &&
 ((DocHasTerm "magnetic pole") || (DocHasTerm "north pole")) &&
 ((DocHasTerm "shifting") || (DocHasTerm "wandering") || (DocHasTerm "shift"))

Constraints 2: (DocHasTerm "earth") &&
 ((DocHasTerm "magnetic pole") || (DocHasTerm "north pole"))

The first constraint, for example, is interpreted by the system as a requirement that the body of the text, in this case a document window, contains the term "earth" and at least one of the terms from each of the two term sets: {"magnetic pole", "north pole"} and {"shifting", "wandering", "shift"}.

The first set of constraints is applied to the result set of the initial query in order to select document windows for feedback. Only those document windows among the top 40 that satisfy the first set of constraints are considered for feedback. The top 50% of the terms in the thesaurus extracted from these document windows are added to the query.

The second set of constraints is applied to the results of the augmented query and the documents that satisfy the constraints are placed at the top of the final retrieval list. The remaining documents that do not satisfy the constraints are included at the bottom of the list

3.2 Comparative Analysis

3.2.1 Results of CLTHES and CLCLUS Experiments

Statistically there is no significant difference in the performance between CLTHES and CLCLUS, as can be seen from Figure 6 and Table 5. However, it is interesting to note that, for a number of queries the precision and recall differ significantly (Figure 7). It is our belief that the observed difference in performance is mostly due to the difference in formulation of the initial NL query, although for some queries the constraints may have caused dramatic changes.

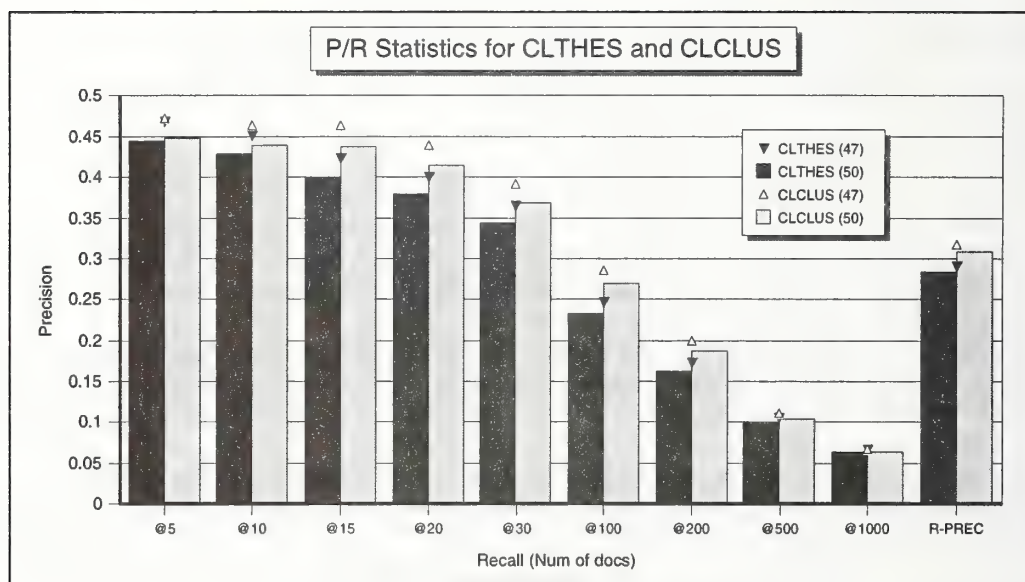
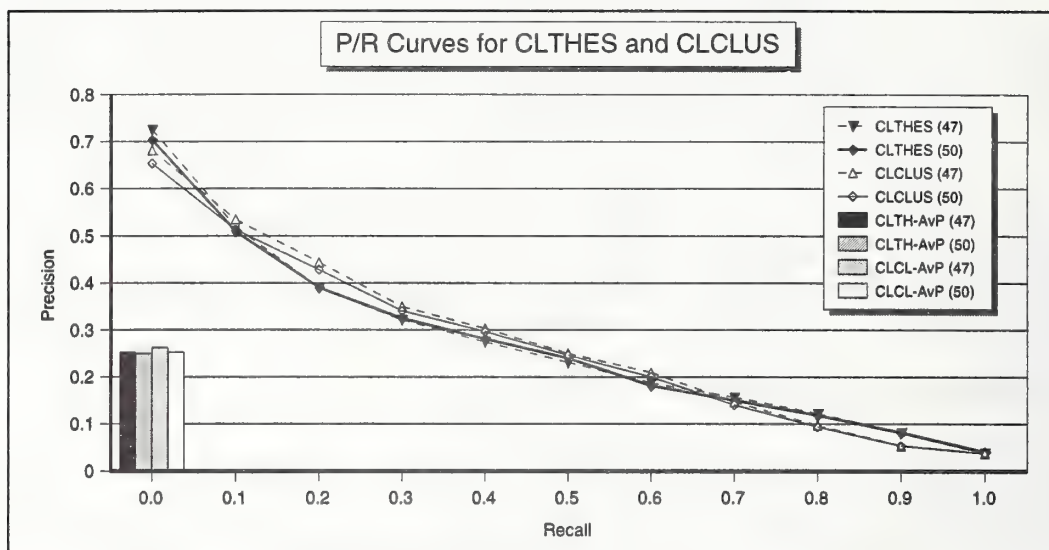


Figure 6: Precision and recall statistics for the CLTHES and CLCLUS Experiments

Experiment	CLTHES (50)	CLTHES (47)	CLCLUS (50)	CLCLUS (47)
Recall	3,147	3,144	3,163	3,160
AvePrecision	0.2513	0.2525	0.2535	0.2629
R_Precision	0.2833	0.2907	0.3085	0.3176
Front-end-prec (at 0.0)	0.7046	0.7247	0.6525	0.6824

Table 5: Performance statistics for CLTHES and CLCLUS

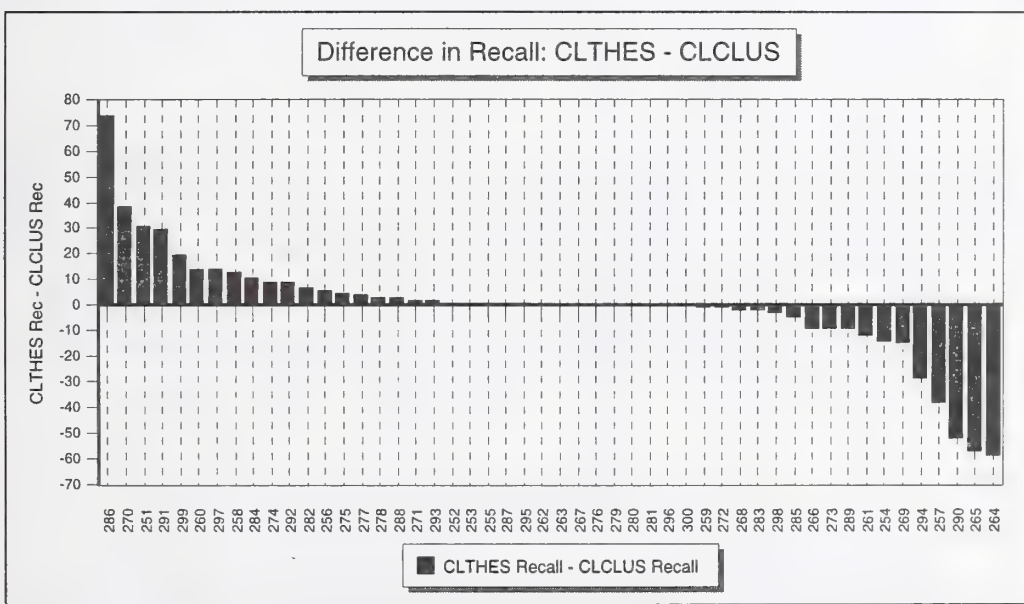
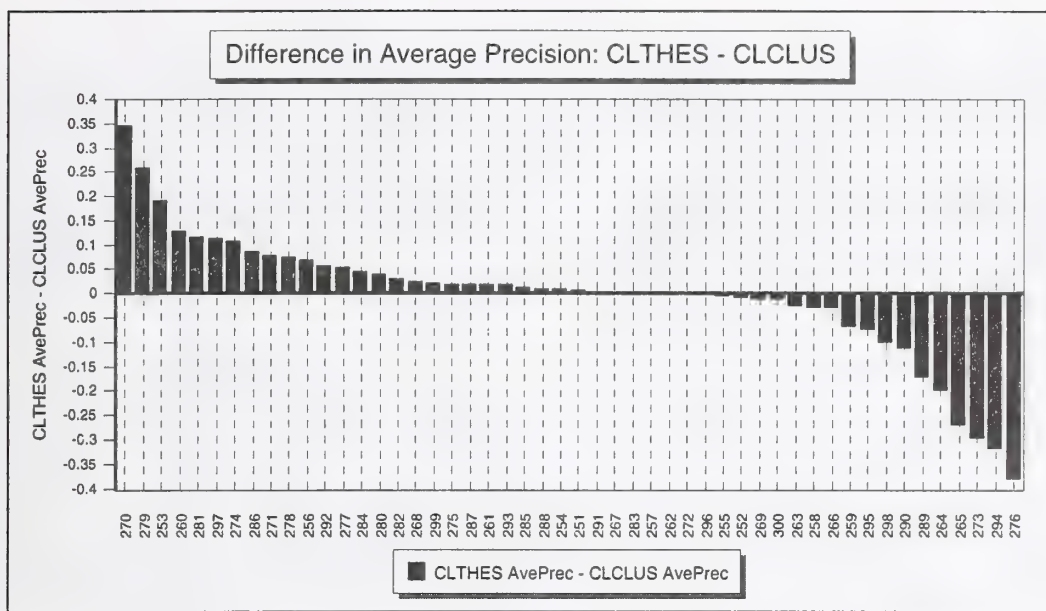


Figure 7: Difference in the retrieval performance for individual queries

Indeed, if we compare the average precision and recall obtained by the NL queries only, without constraints and automatic feedback, the difference in retrieval and average precision statistics has similar characteristics as for CLTHES and CLCLUS experiments (see Figure 8). In fact, for 34 out of 50 topics the relationship between the average precision of CLTHES and CLCLUS initial queries does not change after the controlled feedback and document re-ranking have been applied. For example, the average precision for 19 topics is higher for a CLTHES NL query than for a CLCLUS NL query and it remains such when constraints and the automatic feedback are applied.

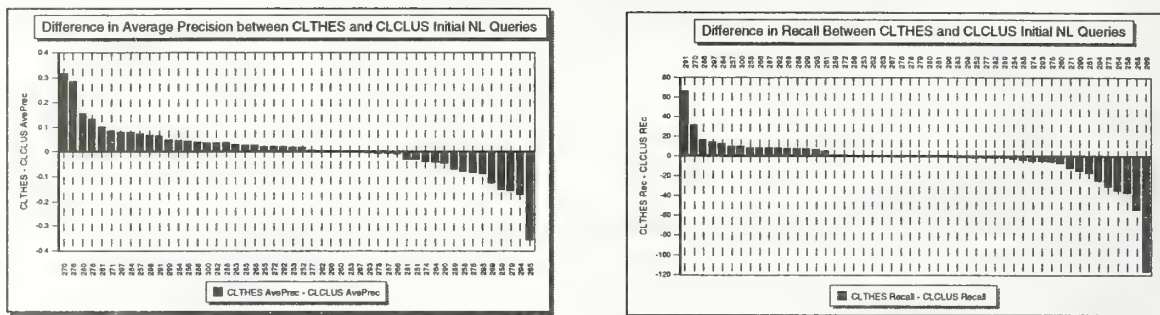


Figure 8: Difference in average precision and recall for CLTHES and CLCLUS initial NL queries

On the other hand, the retrieval performance of some of the queries shows that the constraints may change the average precision quite significantly. For example, for topic 273 we find that the experiment without constraints, thus using only unconstrained automatic feedback, yields average precision of 0.5690. This precision was reduced to 0.2632 when constraints were used to control the feedback and the selection of final documents (see Table 6).

Topic 273: Volcanic and Seismic Activity Levels

	No constr.	Constr.		No constr.	Constr.
Retrieved:	1,000	1,000			
Relevant:	513	513			
Rel_ret:	375	374			
Interpolated Recall - Precision Averages:			Precision:		
at 0.00	1	0.4793	At 5 docs:	0.8	0
at 0.10	0.9091	0.4793	At 10 docs:	0.8	0
at 0.20	0.9035	0.4793	At 15 docs:	0.8667	0
at 0.30	0.8989	0.4793	At 20 docs:	0.9	0.05
Average precision:	0.569	0.2632	R-Precision:	0.5867	0.4581
Constraint:					
(! (DocHasTerm "act")) ¹ (! (DocHasTerm "rule")) &&					
((DocHasTerm "vocanic") ² (DocHasTerm "seismic") (DocHasTerm "earthquake"))					

¹ "!" Indicates Boolean NOT operator ² Misspelled in the experiment

Table 6: Difference in precision between constrained and unconstrained search for Topic 273

3.2.2 Effects of result merging

The final results for both CLTHES and CLCLUS are obtained by merging results of more constrained and less constrained searches, giving the preference to the results from the more constrained search in order to achieve higher front-end precision. Table 7 summarizes the performance statistics of the experiments related to CLTHES.

Experiment	Constraints1	Feedback	Constraints2	Recall	AvePrec	R-Prec	Front-end-prec
E1(Double-Constr.)	yes	yes	yes	2,439	0.233	0.2708	0.6992
I2 (Single-Constr.)	yes	yes	no	3,144	0.2542	0.2933	0.7367
Merge E1 & I2 (CLTHES)				3,144	0.2513	0.2833	0.7046
Merge E1 (100) & I2				3,140	0.2522	0.2864	0.7095

Table 7

Results of the more constrained search E1, which involves two sets of constraints and automatic feedback, are improved by the merge with results from the simple feedback control experiments I2, in which we relax the constraint on the final selection of documents. The recall and all precision statistics are increased. However, when compared with I2 the merged results are inferior, which is the consequence of the significantly worse performance of E1 itself.

In our result merging experiments we also began to explore the effects of different ratios in which the participating list are combined. For example, we used only the top 100 documents from the more constrained experiments and supplemented them with the remaining documents from the less constrained search. While in the case of the 'double constrained' experiment this resulted in a list that is better only than the results of the more constrained search, other experiments indicate that the merged list can attain performance statistics higher than merged components. As an example, we present in Table 8 the same merging procedures for the simple controlled experiments I2 and E2 which involve only the constraints for selection of feedback documents.

Experiment	Constraints1	Feedback	Constraints2	Recall	AvePrec	R-Prec	Front-end-prec
E2 (No-Feedback)	yes	no	no	2,613	0.2455	0.2933	0.7691
I2 (Single-Constr.)	yes	yes	no	3,144	0.2542	0.2933	0.7367
Merge E2 & I2				2,982	0.2626	0.2963	0.7695
Merge E2 (100) & I2				3,144	0.2664	0.303	0.7695

Table 8

3.2.3 Comparison with TREC-5 Participating Systems

In comparison with other systems that participated in the ad-hoc manual category, CLARIT achieved recall and average precision above the median for most of the queries (see Figures 9-10 and Tables 9-10). More precisely, the recall of CLTHES was above or equal to the median for 41 of 50 (82%) queries. Similarly, for CLCLUS, the recall was above or equal to the median for 37 out of 50 (74%) queries. The system achieved the best recall for 11 queries in the CLTHES and for 12 queries in the CLCLUS experiment.

Comparison of the CLARIT average precision for individual queries shows that 38 out of 50 (76%) CLTHES queries and 33 out of 50 (66%) CLCLUS queries achieved precision above or equal to the median. In CLTHES an average precision within 1% from the best average precision was achieved for 4 queries and in CLCLUS for 5 queries. Figures 9-10 show the comparison of the average precision and recall statistics for individual queries.

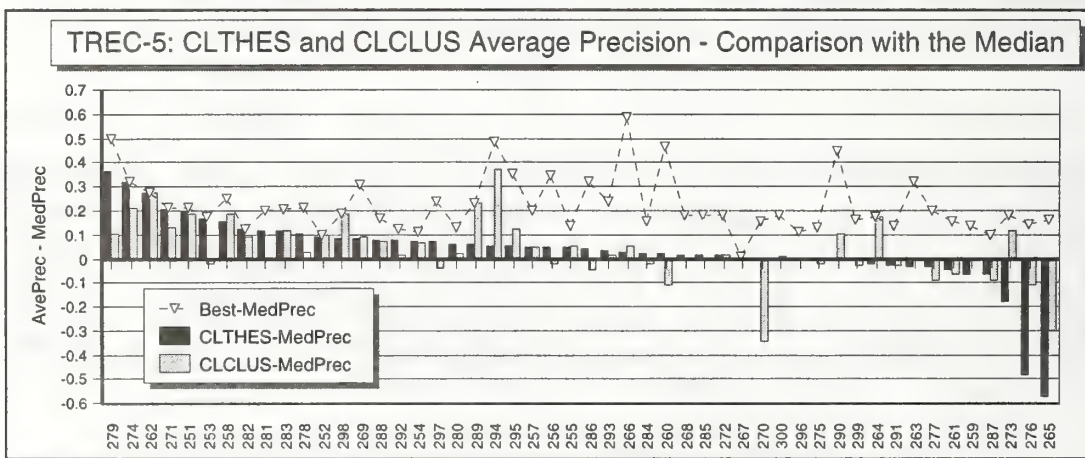
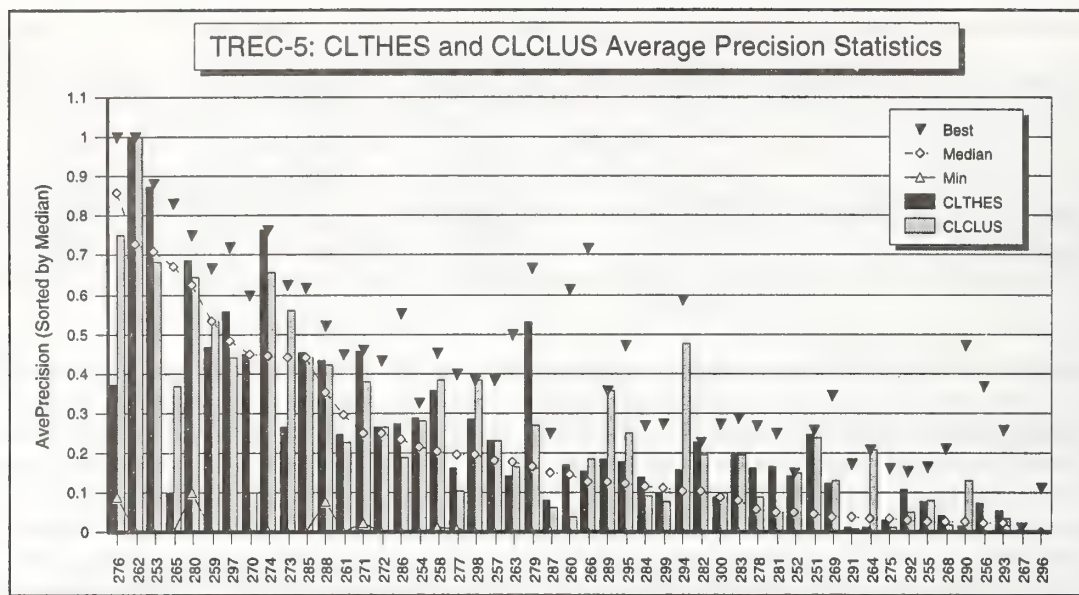


Figure 9: CLTHES and CLCLUS average precision statistics

AvePrec Statistics	> Med	=Med	<Med	Best(1%diff)
CLTHES	33	5	12	4
CLCLUS	28	5	17	12

Table 9: Comparison with the median average precision

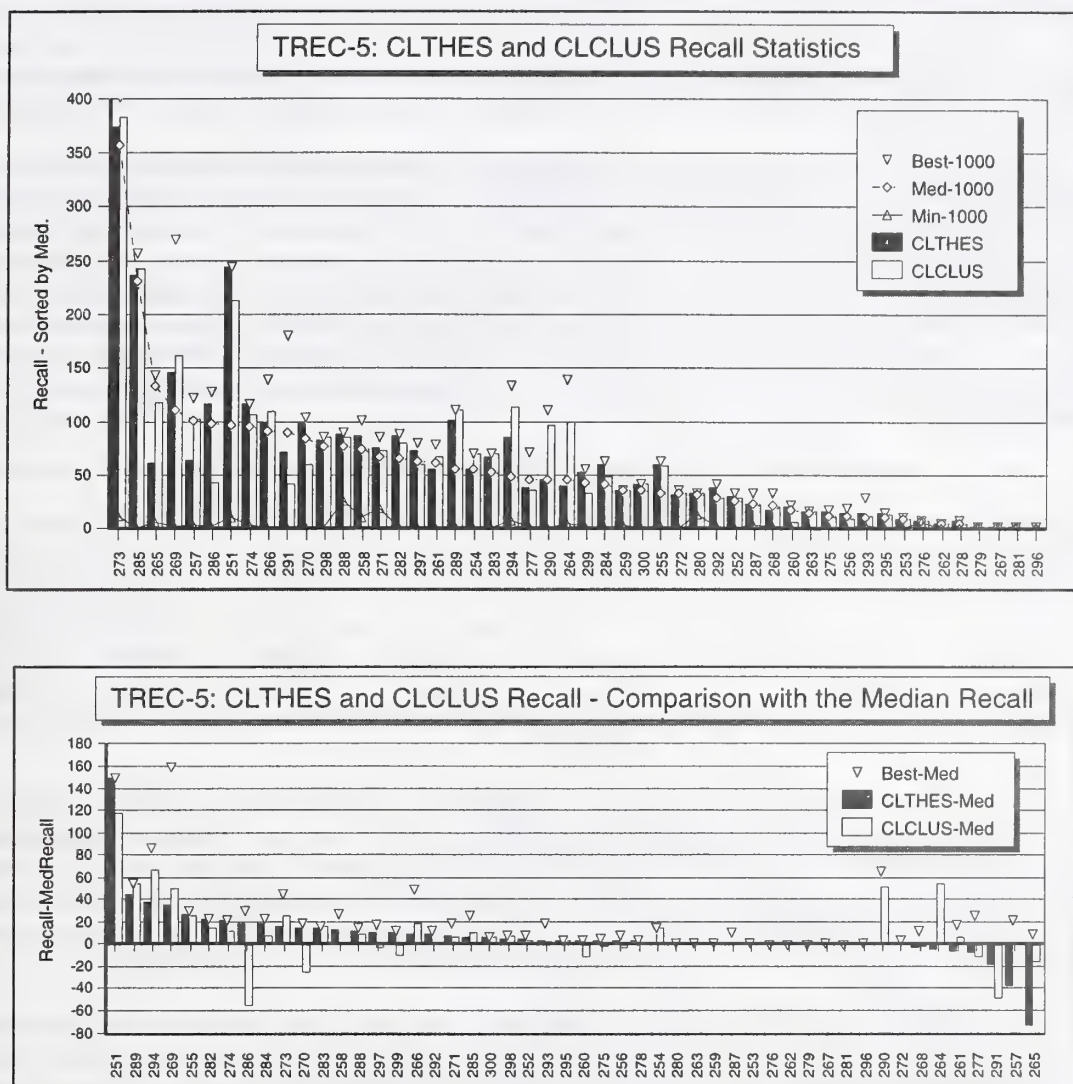


Figure 10: CLTHES and CLCLUS recall statistics

Recall Statistics	> Med	=Med	<Med	Best
CLTHES	32	9	9	11
CLCLUS	27	10	13	12

Table 10: Comparison with the median recall

4. Conclusions

The CLARIT feedback control experiments and the two official TREC-5 runs, CLTHES and CLCLUS, provide useful insights in the interaction between the constraints and automatic feedback. Since the evaluation of experiments is done with respect to available relevance judgment of the NIST experts, the improvement figures reflected in our experiments as reported in this paper are only an approximation of the true performance indicators. Our experience with the analysis of TREC-4 Interactive experiments [6] lead us to believe that the general improvement of retrieval

performance achieved by constraint controlled feedback would be more significant when computed within the actual user's evaluation system. We expect the same to be the case with the results of CLTHES and CLCLUS in which the constraints are also used to facilitate the final ranking of the retrieved documents. We speculate that the slight decrease in performance observed in CLTHES will in fact be interpreted as an improvement in the user's value system since the ranking of documents controlled by user specified constraints more strongly reflects the true user's relevance judgment criteria.

In summary, since the NL queries are created through interactive searches over the target database it is not surprising that the experiment which uses only the initial queries (I0) yields a reasonably high precision (Table 11). Applying fully automatic feedback to the initial queries increases recall significantly at the expense of the precision as can be seen from the retrieval results in the experiment I1. Furthermore, a refined automatic feedback that includes user's input in the form of user constraints helps further improve recall and increase precision. That can be seen from experiment I2. In the experiment CLTHES we observe a slight decrease in precision when user constraints are also used for final document ranking. We believe that this is due to the user bias, in particular to the constraints over-fitting to the documents reviewed and judged relevant by the user during the initial, interactive query formulation process.

Experiment	Constraints1	Feedback	Constraints2	Recall	AvePrec	R-Prec	Front-end-prec
I0	no	no	no	2,944	0.2442	0.2777	0.7648
I1	no	yes	no	3,116	0.2261	0.2548	0.6703
I2	yes	yes	no	3,144	0.2542	0.2933	0.7367
CLTHES	yes	yes	yes	3,144	0.2513	0.2833	0.7046

Table 11

We wish to emphasize that our study of constraint facilitated control feedback is primarily aimed at addressing the problem of limited user access to the target database. We do not expect that the use of constraints will outperform thesaurus based query expansion when the user can manually select relevant documents for automatic feedback. We also do not consider Boolean constraints - as they have been used in classical Boolean systems and in some recent work related to the special IR problems [7] - to be a substitute for the NL representation of a query. Indeed, in our experiments, the NL component of the CLARIT compound query is conceptually much richer than the set of associated constraints.

The constraints used in the final ranking of documents merely ensure that documents which responded to the NL query and contain features that, in user's opinion, should be *sufficient* for characterizing relevant documents, are presented at the top of the list. We assume that a user familiar with the topic has a particular information need that might be better addressed if constraints are used in this manner but by no means would we recommend using constraints as a primary tool for retrieving documents. Indeed, because of the extreme sensitivity of constraints to the global conceptual structure of the database we would not attempt to use user specified constraints as a complete set of sufficient conditions for determining document relevance.

References

- [1] Evans, David A. and Lefferts, Robert G. "Design and Evaluation of the CLARIT-TREC-2 system". In Harman D. (Ed.), *The Second Text REtrieval Conference (TREC-2)*. National Institute of Standards and Technology Special Publication 500-215. Washington, DC: Government Printing Office, 1994, 137-150.
- [2] Evans, David A., Milic-Frayling, Natasa and Lefferts, Robert G. "CLARIT TREC-4 Experiments". In Harman D. (Ed.), *The Fourth Text REtrieval Conference (TREC-4)*. NIST Special Publication. Washington, DC: U.S. Government Printing Office. 305-321.

- [3] Harman D. (Ed.), (1993). *The First Text REtrieval Conference (TREC-1)*. National Institute of Standards and Technology Special Publication 500-207, Gaithersburg, Md. 20899.
- [4] Harman D. (Ed.), (1994a). *The Second Text REtrieval Conference (TREC-2)*. National Institute of Standards and Technology Special Publication 500-215, Gaithersburg, Md. 20899.
- [5] Harman D. (Ed.), (1995). *The Third Text REtrieval Conference (TREC-3)*. National Institute of Standards and Technology Special Publication 500-225, Gaithersburg, Md. 20899.
- [6] Milic-Frayling, Natasa, Zhai, C., Tong, X., Mastroianni, M.P., Evans, D.A., and Lefferts, R.G. "CLARIT TREC-4 Interactive Experiments". In Harman D. (Ed.), *The Fourth Text REtrieval Conference (TREC-4)*. NIST Special Publication. Washington, DC: U.S. Government Printing Office. 323-357.
- [7] Hearst, Marti A. "Improving Full-Text Precision on Short Queries using Simple Constraints". Proceedings of the Fifth SDAIR 96, 217-232, Las Vegas, Nevada, 1996.

Appendix

CLARIT TREC-5 Very Large Collection Experiments

CLARIT Very Large Collection (VLC) Experiments were performed with the goal of exploring the efficiency and effectiveness of the CLARIT IR System over large collections of data. In order to complete the task we indexed all the individual databases separately. The unique retrieval list was obtained by merging the results from retrieval over individual databases.

We submitted two runs for each task, the baseline and the main task:

CLVLCBA	- baseline (2G) with automatically created queries
CLVLCBC	- baseline (2G) with automatically queries + manually created constraints
CLVLCMA	- main (4G) with automatically created queries
CLVLCMC	- main (4G) with automatically created queries + manually created constraints.

In all four runs, the queries were constructed by applying CLARIT NL processing to the full description of the TREC topics. In the runs with the constraints, CLVLCBC and CLVLCMC, we applied Boolean type filters to the vector space results of the initial queries to select documents for automatic feedback. In these experiments we used the same set of constraints that were designed for the CLARIT ad-hoc experiments, CLTHES and CLCLUS.

Evaluation of the VLC track experiments is based on the precision achieved at 20 retrieved documents. According to the answer key for the VLC track, CLARIT system performance is as follows:

Task	Precision at 20 Documents	
	NL Queries	NL Queries + Constraints
Baseline	CLVLCBA: 0.2012	CLVLCBC: 0.2292
Main Task	CLVLCMA: 0.3083	CLVLCMC: 0.492

Table 1

For the experiments we used a DECAlpha workstation with 128M of RAM. In Table 2 we summarize the system characteristics and performance statistics.

		Main Task	Baseline Task	Ratio (Main/Baseline)
Data Structure Building Speed (Mb/hour)		200	200	1
Disk Space Requirement		8G	4G	2
Memory Requirements (for retrieval)		32M	22M	1.45
Query Processing Speed (No. of queries/hour)	NL Queries	5.78	13.5	0.428
	NL Queries + Constraints	4.82	11.43	0.422

Table 2

Experiments on Chinese Text Indexing —CLARIT TREC-5 Chinese Track Report

Xiang Tong¹, ChengXiang Zhai¹, Nataša Milić-Frayling², and David A. Evans²

¹ Laboratory for Computational Linguistics
Carnegie Mellon University
Pittsburgh, PA 15213
{xt22@andrew.cmu.edu, cz25@andrew.cmu.edu}

² CLARITECH Corporation
5301 Fifth Ave.
Pittsburgh, PA 15232-2124
{natasa@clarit.com, dae@clarit.com}

1 Introduction

The focus of the CLARIT^{TM1} Chinese Track Experiments is on investigating the effectiveness of different automatic indexing methods for retrieval over Chinese texts. In particular, we explored indexing using linguistic units (words, compound words, and phrases), single Chinese characters, and overlapping character bigrams. In addition to fully automatic processing of queries, we ran experiments with manually constructed term vector queries supplemented by Boolean type constraints. The constraints were used for selecting documents for CLARIT automatic feedback or as a mean of refining the final set of retrieved documents [Milić-Frayling et al. 1997].

All the experiments were conducted using the CLARIT retrieval system [Evans & Lefferts 1995]. Since its current NLP component does not support the parsing of Chinese texts, we designed an appropriate parsing module and pre-processed the documents before submitting them for indexing and retrieval by the CLARIT system.

2 Chinese Text Indexing and Retrieval Experiments

In this section, we describe the two main indexing methods we explored: lexical term indexing and character N-gram indexing. The lexical term indexing method represents a linguistic approach while the N-gram indexing method is a purely statistical approach to indexing of Chinese texts. We also briefly discuss the general procedure and the design of our automatic and manual experiments.

2.1 Indexing by Lexical Terms

In contrast to English texts, Chinese texts have no obvious word boundaries. The text consists of a sequence of non-spaced ideographic characters, whose closest equivalents in English are morphemes. These morphemes normally are meaningful and represent words. But they can also be grouped together into multi-character units forming non-decomposable compound words. The problem of identifying words in Chinese text is very

¹CLARIT is a registered trademark of CLARITECH Corporation

similar to that of identifying multi-word lexical items (or non-decomposable phrases) in English text, since the criteria for determining such English phrases is also not easy to specify.

We segment the Chinese text via a two-phase procedure using various lexical resources and heuristics.

In the first phase, we segment the text into lexical units by applying

- lexical lookup using a Chinese lexicon that contains about 100,000 entries including words, compounds, and phrases extracted from several Chinese on-line dictionaries,
- a dynamic programming algorithm that optimizes the segmentation of a sentence into lexical words by searching for the segmentation with the minimum number of words,
- simple morphological rules, and
- heuristics about Chinese names, transliterations of foreign names, and other new compound words.

Portions of the segmented text that contain consecutive characters that are not part of any compounds or phrases are analyzed by applying heuristics for discovering new words. These items are collected into a list of unknown terms for further analysis and clean-up. After the whole corpus is processed, an item in the list is removed if it has a frequency lower than a pre-specified threshold or contains very frequent function words. This procedure when applied to the TREC Chinese data leads to a new word list of about 76,000 entries that are then incorporated into the original lexicon to form a new augmented lexicon with a total of approximately 176,000 entries.

In the second phase, the same dynamic algorithm and morphological rules are used to segment the texts into terms in the augmented lexicon. Moreover, all text segments with more than two characters are further decomposed into subcomponents that are valid words in the dictionary. For example, if *A, ..., G* are Chinese characters and *ABCD, AB, CD, DEF, DE*, and *G* are valid words in the augmented lexicon, the sentence *ABCDEFGH* may be converted into *ABCD<space>AB<space>CD<space>DEF<space>DE <space>G*. All text segments and their valid subcomponents are used to represent the content of the text. One-character function words are not used for indexing, because they are highly frequent in the text and thus have low discrimination value.

After the text has been pre-processed by the text segmentation module and converted into lexical items separated by spaces, the converted text is submitted to the CLARIT indexer, which treats these lexical items as single terms.

In the Chinese automatic experiments, we constructed queries automatically, using all the parts of the topic description, after they were preprocessed by the same segmentation module that processed documents. The task involved 28 Chinese topics, each of which contains three parts: the title, topic description, and the narrative. The terms in each section were assigned a specific weight: terms in the title a weight of 2, terms in the description a weight of 4, and terms in the narrative a weight of 1.

In the manual runs, the automatically created queries were manually modified by adjusting the term weights and adding new terms. We also specified Boolean type constraints to select documents for automatic feedback. Documents that were retrieved in response to the initial query and satisfied the constraints were considered useful for automatic feedback. The constraints were written as a set of Boolean expressions using AND, OR, and NOT operators [Evans et al. 1996] [Milić-Frayling et al. 1997].

Manual modification of the queries took about 4 hours (5 min/query) and it was performed by a native speaker of Chinese. On average, 2–3 terms were added to a query.

2.2 Indexing with Overlapping Character N-grams

An alternative way to index Chinese text is to use character N-grams as indexing units. In our TREC-5 experiments we implemented and evaluated indexes based on unigrams (i.e., single characters) and overlapping character bigrams.

In our N-gram indexing approach, each sentence in the text is first converted into overlapping N-grams that are separated by spaces. For instance, indexing based on overlapping bigrams would convert a sequence of Chinese characters *ABCDEFGH* into a sequence of bigrams *AB<space>BC<space>CD<space>DE<space>EF<space>FG*. Text in this form is then processed by the CLARIT indexer.

Similar to the document text, queries are converted automatically into overlapping N-grams. The resulting character N-grams are weighted based on the section of the topic they originate from (see Section

2.1). We did not attempt any manual modification of the N-gram queries. Thus, all the runs that use this indexing method are considered TREC automatic runs.

3 Results and Analysis

The document collection used for the TREC-5 Chinese track consists of news articles from *People's Daily* and *Xinhua News Agency*. The size of the corpus is about 170 MB. We present the evaluation of our results based on the 19 topics that had been assessed at the time of TREC-5 meeting (November 1996).

The two official runs we submitted, CLCHNA and CLCHNM, use lexical term indexing. The difference between the two runs is in the construction of queries: CLCHNA uses automatically created queries, while CLCHNM uses manually modified queries.

We performed a number of experiments in our evaluation of different indexing methods. All the experiments with automatic feedback involve the query expansion based on the CLARIT thesaurus extraction technique [Evans & Lefferts 1995]. The results from the automatic runs using different indexing terms (characters, character bigrams, and lexical units) are shown in Table 1.

All manual runs were conducted over the database indexed by lexical terms. The results are presented in Table 2. In Figure 1 we show the precision-recall curves for all the runs.

Table 1: Automatic Run Results for Different Indexing Methods

Runs	Rcl(Rel Docs)	Ave Pre	Init Prec	R-Prec
Character-NoFeedback(baseline)	0.5318(744)	0.1235	0.3588	0.1502
Character-Feedback increase over baseline	0.3345(468) -37.1%	0.0080 -93%	0.3133 -12.7%	0.1100 -26.8%
Bigram-NoFeedback increase over baseline	0.9078(1270) +70.7%	0.2541 +100.5%	0.6060 +68.9%	0.2877 +91.5%
Bigram-Feedback increase over baseline	0.8549(1196) +60.8%	0.2479 +100.7%	0.6535 +82.1%	0.2963 +97.3%
Lexicon-NoFeedback increase over baseline	0.8670(1213) +63.0%	0.2586 +109.4%	0.6674 +86.0%	0.2788 +85.6%
Lexicon-Feedback(CLCHNA) increase over baseline	0.8449(1182) +59.8%	0.2677 +116.8%	0.6223 +73.4%	0.2998 +99.6%

Table 2: Manual Run Results

Runs	Rcl(Rel Docs)	Ave Prec	Init Prec	R-Prec
Manual-NoConstraint-NoFeedback increase over baseline	0.8863(1240) +66.7%	0.3079 +149.3%	0.7758 +116.2%	0.3296 +119.4%
Manual-NoConstraint-Feedback increase over baseline	0.84(1175) +58.0%	0.2921 +136.5%	0.7613 +112.2%	0.3371 +124.4%
Manual-Constraint-NoFeedback increase over baseline increase over NoConstraint-NoFeedback	0.8956(1253) +68.4% +1.0%	0.3855 +212.1% +25.2%	0.8383 +133.6% +8.0%	0.4122 +174.4% +25.1%
Manual-Constraint-Feedback(CLCHNM) increase over baseline increase over NoConstraint-Feedback	0.8956(1253) +68.4% +6.6%	0.3583 +190.0% +22.7%	0.7534 +110.0% -1.0%	0.3872 +157.8% +14.9%

From the collected performance statistics we can conclude that the results from the single character indexing are significantly worse than those from the experiments with other indexing methods. This indicates that Chinese characters have low discrimination value since they tend to be too general and ambiguous.

On the other hand, the retrieval performance based on overlapping character bigram indexing is comparable to that based on lexical term indexing. This suggests that the character bigrams are useful features in

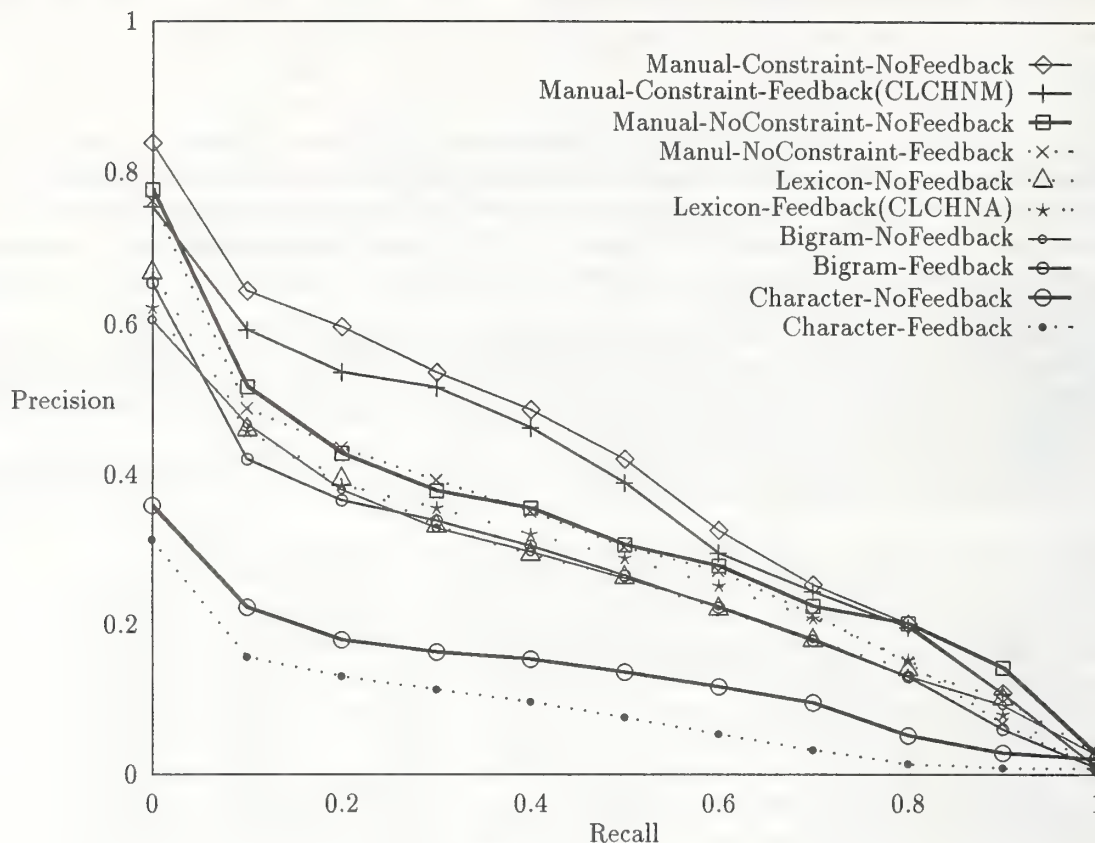


Figure 1: Precision-Recall Curves for All the Runs

capturing the patterns of Chinese texts, even though a large number of bigrams are not meaningful linguistic units.

Among the four manual runs, those that used Boolean constraints to control feedback achieved more than 20% improvement in average precision over the corresponding runs without constraints. Thus, Boolean constraints seem to be very effective in searching over Chinese texts.

The effect of automatic query augmentation based on the CLARIT thesaurus extraction technique is unclear at this point. Contrary to our past experience with retrieval over English texts, the feedback applied to manual runs hurts both the average and the initial precision. In fact, the best average and initial precision are achieved in the manual run using constraints and no feedback. We suspect that the present CLARIT thesaurus extraction technique is not quite appropriate for Chinese text. However, this issue needs to be explored further.

4 Conclusions

In the CLARIT TREC-5 Chinese experiments, we evaluated the effectiveness of different indexing methods for Chinese text retrieval. We found that indexing based only on single Chinese characters gives poor retrieval performance because the meaning of most Chinese characters tends to be general and ambiguous. On the other hand, indexing on overlapping character bigrams leads to results comparable to the retrieval based on lexical term indexing.

In the manual runs using lexical term indexing, Boolean type constraints proved to be very useful in specifying user information needs and therefore improve the retrieval performance.

In our future work we will continue investigating existing and new methods for automatic expansion of Chinese queries. Furthermore, we will explore the effectiveness of additional character N-gram indexing techniques.

References

- [Evans & Lefferts 1995] Evans, D. A. and Lefferts, R. G. 1995. CLARIT-TREC Experiments. *Information Processing and Management*, Vol. 31, No. 3, 385-395.
- [Evans et al. 1996] Evans, D. A., Milić-Frayling, N., Lefferts, R. G. 1996. CLARIT TREC-4 Experiments. In: Harman, D.(Ed.), *The Fourth Text REtrieval Conference (TREC-4)*. NIST Special Publication 500-236, 1996, 305-321.
- [Milić-Frayling et al. 1997] Milić-Frayling, N., Tong, X., Zhai, C., and Evans, D. A. 1997 CLARIT Compound Queries and Constraint-controlled Feedback in TREC-5 Ad Hoc Experiments. In: Harman, D.(Ed.), *The Fifth Text REtrieval Conference (TREC-5)*. NIST Special Publication, 1997. (This volume.)



OCR Correction and Query Expansion for Retrieval on OCR Data —CLARIT TREC-5 Confusion Track Report

Xiang Tong¹, ChengXiang Zhai¹, Nataša Milić-Frayling², and David A. Evans²

¹ Laboratory for Computational Linguistics
Carnegie Mellon University
Pittsburgh, PA 15213
{xt22@andrew.cmu.edu, cz25@andrew.cmu.edu}

² CLARITECH Corporation
5301 Fifth Ave.
Pittsburgh, PA 15232-2124
{natasa@clarit.com, dae@clarit.com}

1 Introduction

In CLARIT^{TM1} TREC-5 confusion track experiments, we explored two techniques for improving retrieval performance over corrupted data: (1) OCR word error correction to improve OCR text accuracy, and (2) query expansion by adding query term variants found in the corrupted text. The OCR word correction technique is based on statistical word bigram modeling [Tong & Evans 1996]. The variants of a query term are terms similar to the query term, as measured by the edit distance [Wagner 1974]. While the official runs were based on the first approach, in our follow-up experiments we tested the second approach as well.

In this report we first give a brief description of the OCR correction and query expansion techniques, and then discuss the results of our experiments.

2 The Automatic OCR Correction System

The OCR correction technique used in the CLARIT confusion experiments processes the whole text, sentence by sentence. Figure 1 shows the architecture of the system. For a given text, the correction procedure has three main steps:

1. Read a sentence from the OCR text.
2. Retrieve up to M candidates from the lexicon for each detected error that does not match any lexicon entry. Rank the M candidates by their conditional probabilities with regard to the error. Keep only the top N candidates for the next processing step.
3. Use the Viterbi algorithm to get the maximum likelihood word sequence for the sentence.

In the experiments reported in [Tong & Evans 1996], this method corrected up to 50% of errors in the OCR texts, when applied with $M = 2000$ and $N = 10$. In our TREC-5 confusion experiments we used $M = 200$ and $N = 10$ in order to speed up the correction process. We were aware that this parameter setting may result in a performance inferior to the one observed in earlier experiments.

¹CLARIT is a registered trademark of CLARITECH Corporation

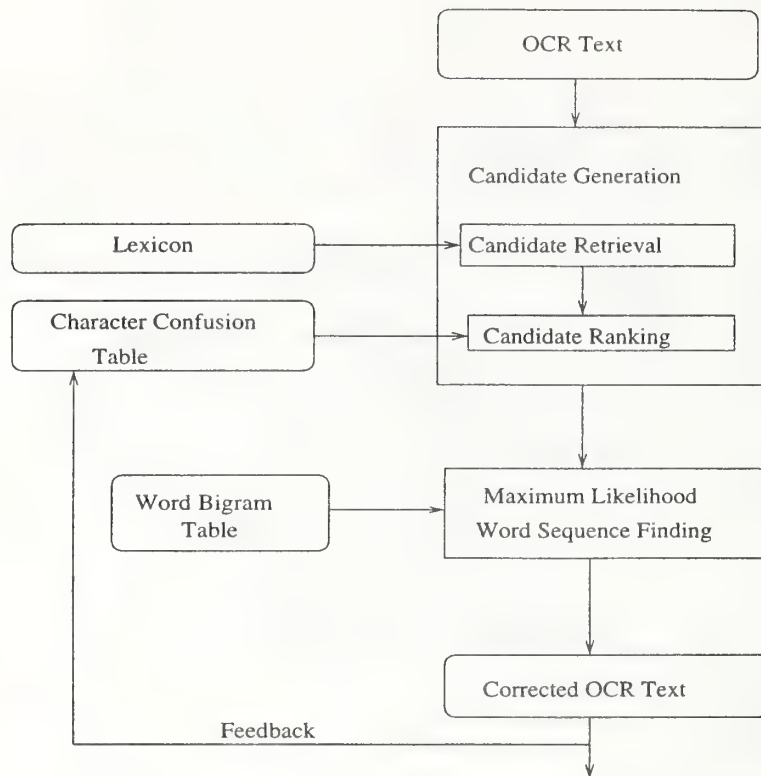


Figure 1: OCR correction system architecture

3 Query Expansion

The original query is expanded by adding variants from the corpus that are similar to the query terms as measured by edit distance [Wagner 1974]. For each query term, at most 10 variants within edit distance of 3 are added. The original query term is assigned a weight of 9 and each of the variants is assigned a weight of $(4 - \text{edit_distance})$. This method is similar to the query expansion performed by Cornell in TREC 4 [Buckley et al. 1996]. The main difference between the two methods is that the Cornell group expanded queries with all variants with edit distance 1, and used the combined *idf* scores of original and added terms to weight the query terms.

The following is an example of query expansion for Topic 36. There is only one word, “headband”, in the original query.

Topic 36: headband

Final vector after expansion:

```

<t cf="9">headband</>
<t cf="2">headbands1</>
<t cf="2">headboard</>
<t cf="2">headend</>
<t cf="3">headland</>
<t cf="2">headlands</>
<t cf="2">headpond</>
<t cf="2">headwind</>
<t cf="2">hedland</>
<t cf="2">leadand</>
<t cf="2">neadland</>

```


4 Experiments and Results

The TREC-5 confusion task was named “poorly identified known item search”. It involved 49 queries, each uniquely identifying a single document in the FR94 database. The data for searching are provided in three forms: the original uncorrupted data, slightly degraded OCR data with 5% word error (degrade5), and significantly degraded OCR data with 20% word error (degrade20). A participating group is allowed to submit up to 1,000 documents for each topic. The rank of the target item in the retrieved list is recorded. If the known item is not retrieved, a rank of 2000 is assigned. The performance of the system is evaluated using average expected search length and mean reciprocal, which are the mean rank and the mean of the reciprocal of the rank calculated over all the topics respectively.

4.1 Official Runs

In the CLARIT official runs, the two degraded databases were first corrected by our OCR correction module and then indexed using the standard CLARIT indexing module. We used Federal Register data from 1988 and 1989 as the training data to create a lexicon and estimate word uni-gram and bi-gram probabilities. The correction was run on a DecAlpha workstation with 128 MB RAM and it took about 5 days to correct each of the two 250M degraded databases.

The original natural language queries were submitted to the CLARIT retrieval system, which parsed the queries into single terms and phrases.

For comparison purposes we performed the retrieval experiments over all three sets of data: uncorrupted, degrade5, and degrade20 data. Search over the uncorrupted data was performed using the original queries without automatic feedback. In the experiments over degrade5 and degrade20 data, we evaluated system performance for both the initial queries and the enhanced queries from automatic feedback. The results of the simple initial retrieval (CLCON5 and CLCON20) and those with the automatic feedback (CLCON5F and CLCON20F) are in Table 1. Figure 2 shows the number of documents retrieved at different ranks. Table 2 contains the performance based on the results of all the participating groups.

Table 1: Results of all the CLARIT official runs

	Uncorrupted	CLCON5	CLCON5F	CLCON20	CLCON20F
Average Rank	9.39	84.92	66.82	282.16	286.694
Mean Reciprocal	0.73	0.40	0.23	0.21	0.19
Rank 1-10	44	30	24	21	21
Rank 11-100	4	15	21	15	13
Rank 101-1000	1	3	3	8	10
Not Found	0	1	1	5	5

Table 2: Best, worst, and median average rank and mean reciprocal scores across all the groups

	Uncorrupted			Degrade5			Degrade20		
	Best	Med	Worst	Best	Med	Worst	Best	Med	Worst
Average Rank	8.24	89.18	228.49	25.10	116.73	776.55	115.41	280.78	837.51
Mean Reciprocal	.74	.39	.20	.57	.30	.19	.50	.21	.12

Our performance on uncorrupted data is very close to the best systems in terms of both the expected run length and mean reciprocal. The performance over the slightly degraded data (CLCON5) is better than the median.

For both CLCON20 and CLCON20F, the evaluation statistics are worse than the median. We suspected degrade20 text contains many word errors involving space boundaries. Indeed, there are numerous run-on (“of the”/“ofthe”) and split-word (“training”/“train ng”) errors. The current OCR correction method can not deal with such boundary errors.

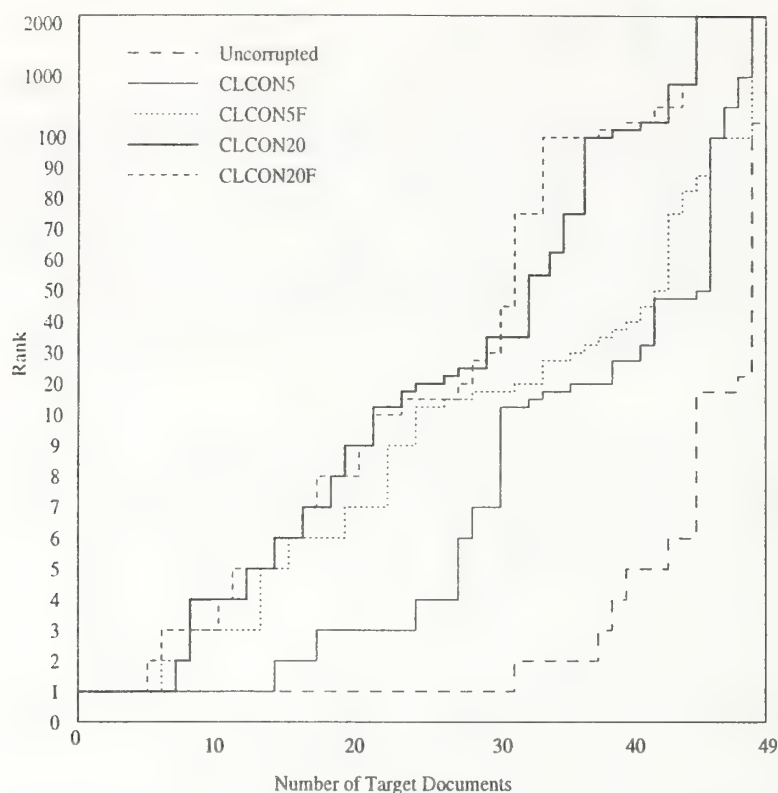


Figure 2: Number of target documents retrieved at a given rank for all the official runs

4.2 Follow-up Experiments on Degrade5 Data

In TREC-5 follow-up experiments, we tested the query expansion technique and compared it with the OCR correction approach. These experiments were run on Degrade5 data. In contrast to the official runs, in these experiments, we did not use phrases to represent queries but only single words. Table 3 contains the retrieval results and Figure 3 shows in detail the number of target documents retrieved at different ranks.

Table 3: Results from follow-up experiments on degrade5 data

Database	uncorrupted	uncorrected		corrected	
Query	unexpanded	unexpanded	expanded	unexpanded	expanded
Ave Rank	5.69	143.4	27.8	80.1	66.0
Mean Rec	0.77	0.618	0.556	0.456	0.376
1-10	46	39	38	32	31
11-100	2	4	10	13	12
101-1000	1	3	1	3	5
Not Found	0	3	0	1	1

The degrade5 baseline run, which uses the unexpanded queries over the uncorrected database, has the best mean reciprocal among all the degrade5 runs. For 35 queries, the target documents are among the top 3 documents in the rank lists. This indicates that CLARIT retrieval system is robust enough to retrieve the relevant documents based on query terms that remain uncorrupted in the target documents. The system is not able to retrieve the target documents for only 3 out of the 49 queries.

Using expanded queries over the original database, the system finds all the target documents with the mean reciprocal 0.556 only a slightly worse than the baseline 0.618 (see Table 3). Furthermore, all but 1 of

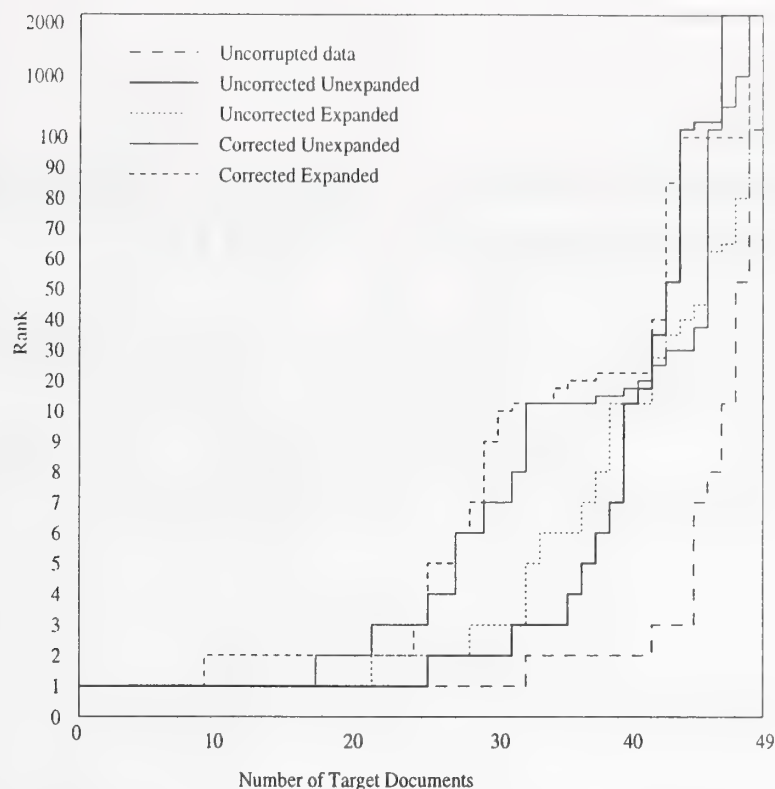


Figure 3: Number of target documents retrieved at a given rank for all the follow-up experiments

these known items are retrieved within rank 100. This shows that our query expansion technique is effective for solving the problem of mismatch between clean queries and corrupted data.

The effectiveness of our OCR correction method is not clear in these experiments, although it is worth noticing that both runs over the corrected degraded5 database miss only one target document each.

5 Conclusions

In TREC-5 confusion experiments, we tested OCR word error correction and query expansion approaches to enhance the retrieval on OCR texts. The corpus-based query expansion technique was quite effective; the OCR correction technique needs to be investigated further.

References

- [Buckley et al. 1996] Buckley, C., Singhal, A., Mitra, M., and Salton, G. 1996. New Retrieval Approaches Using SMART: TREC 4. In: D.K Harman, (Ed), *Proceeding of the Fourth Text REtrieval Conference (TREC-4)*. NIST Special Publication 500-236, 1996, 25-48.
- [Tong & Evans 1996] Tong, X. and Evans, D. A. 1996. A Statistical Approach to Automatic OCR Error Correction in Context. In: *Proceedings of the Fourth Workshop on Very Large Corpora*, 88-100. Copenhagen, Denmark, 1996.
- [Wagner 1974] Wagner, R. A. 1974. The String-to-string Correction Problem. *J. ACM*, 21, 1, Jan. 1974, 168-173.



Evaluation of Syntactic Phrase Indexing — CLARIT NLP Track Report

Chengxiang Zhai¹, Xiang Tong¹, Nataša Milić-Frayling², David A. Evans²

¹ Laboratory for Computational Linguistics
Carnegie Mellon University
Pittsburgh, PA 15213
{cz25@andrew.cmu.edu, xt22@andrew.cmu.edu}

² CLARITECH Corporation
5301 Fifth Ave.
Pittsburgh, PA 15232-2124
{natasa@clarit.com, dae@clarit.com}

1 Introduction

The CLARIT^{TM1} NLP track effort is focused on evaluating the usefulness of syntactic phrases for document indexing. The CLARIT system has several NLP techniques integrated with the vector space retrieval model [Evans et al. 91, Evans et al. 95]. The NLP techniques used in CLARIT include morphological analysis, robust noun-phrase parsing, and automatic construction of first order thesauri, among others. One main feature of CLARIT indexing is that it emphasizes phrase-based indexing with different options for decomposing noun phrases into smaller constituents, including single words. In past TRECs, the default mode for indexing involved full noun phrases, single words, and occasionally selected sub-phrases.² While some early experiments have shown the effectiveness of noun phrases for indexing [Evans et al. 91], there is no direct evaluation of their effectiveness in the context of TREC. In particular, the contribution of small sub-phrases to retrieval performance has not been evaluated outside the context of overall system performance.

The version of the CLARIT system that we used in the experiments has its NLP component tightly integrated with the rest of the system. This does not allow easy evaluation of the individual NLP components. We, therefore, developed separate NLP modules and used CLARIT as a retrieval engine only to evaluate the effectiveness of phrase-based indexing.

2 Phrases for indexing

Most current IR systems use word-based indexing often supplemented with phrases that are obtained using simple statistical approaches, such as a frequency counting of adjacent word pairs. Intuitively, it seems that syntactic phrases (i.e., phrases with certain syntactic relations) can represent document content more precisely than single words and phrases discovered using simple statistical methods, and are thus potentially useful for indexing [Evans et al. 91, Evans et al. 95, Strzalkowski et al. 95, Evans & Zhai 96, Hull et al. 96, Strzalkowski et al. 97].

Single words, as indexing units, may have two different kinds of problems:

¹CLARIT is a registered trademark of CLARITECH Corporation.

²In the TREC-4 ad hoc task, attested subcompounds were used in addition to the full noun phrase and single words.

1. They may be misleading.

In the context of lexical atoms,³ such as “hot dog”, the contained single words do not carry their regular meanings, and are thus very misleading if used as separate indexing terms;

2. They may be too general.

For example, the individual words “junior” and “college” are not specific enough to distinguish “college junior” from “junior college”.

In the first case, it is desirable to use the phrase as a whole nonseparable unit for indexing and not to include the constituent single words in the index. In the second case, it is desirable to supplement single words with more specific and discriminative phrases. The use of simple phrases as additional indexing terms addresses the second problem. While it is very interesting and important to compare syntactic phrases with such simple phrases (cf., for example, [Fagan 87, Hull et al. 96, Strzalkowski et al. 97], we are more interested in evaluating different kinds of syntactic phrases. Based on these observations, we are inclined to propose the following two hypotheses:

1. The use of lexical atoms, such as “hot dog”, to replace single words for indexing would increase both precision and recall;
2. The use of syntactic phrases, such as “junior college” to supplement single words would increase precision without hurting recall and using more such phrases results in greater improvement in precision.

The goal of our efforts with the NLP track is to test these two hypotheses.

In the literature, syntactic phrases have been reported to show no significant improvement in retrieval performance (cf., for example, [Lewis 91, Belkin & Croft 87, Fagan 87], among others). However, the size of the collection used in early experiments is relatively small. It is important to determine whether a larger size collection will make a difference.

3 Experiment Design

3.1 Lexical atom experiment

The goal of lexical atom experiments is to test the effect of taking adjacent word pairs such as “hot dog” as non-separable units, or lexical atoms, for indexing. Single words would not be included in the index unless they occur in the document as single word noun phrases, or as part of decomposable phrases (i.e., phrases that are not lexical atoms). This is expected to have an impact on the retrieval performance by eliminating the possible false matching with single words that form a lexical atom.

Lexical atoms are identified automatically by the two heuristics tests based on the following observations: (1) words in lexical atoms usually have strong fixed association, and thus tend to co-occur as a phrase and (2) when the words in a lexical atom co-occur in a noun phrase, they are never or rarely separated. Intuitively, we are looking for those high frequency word pairs that tend not to be separated by other words within the context of noun phrases. For the experiments reported in this paper, we only considered the pairs formed by two nouns or one adjective followed by a noun. A detailed description of the method can be found in [Evans & Zhai 96].

To test the effect of lexical atoms for any particular configuration of the system, we add the lexical atoms to the CLARIT lexicon, and specify that they should be treated as a non-separable indexing unit. The database and the queries are then processed using the extended lexicon to prevent lexical atoms from being decomposed. The results can be compared with a baseline obtained by using the standard CLARIT lexicon⁴ to index the database and process the queries. Since the lexicon is the only difference, any performance difference will reflect the effect of the lexical atoms added to the lexicon.

³It is hard to give a strict definition for lexical atoms, but it generally refers to a fixed or “sticky” phrase where at least one constituent word’s regular meaning is quite different from the meaning of the phrase.

⁴The lexicon has currently about 80,000 lexical items including a few common proper names.

3.2 Phrase combination experiments

The goal of phrase combination experiments is to test the effectiveness of using syntactic phrases to supplement single words for indexing. Specifically, we want to see if adding phrases to the index will improve the performance significantly and to see how the performance changes according to different combinations of phrases.

The general procedure of the phrase combination experiments is described in Figure 1. We first use a separately developed phrase extraction module to generate different sets of indexing terms including both single words and phrases, and then make the CLARIT system treat every term we generated as a non-decomposable noun phrase. Basically we “force” the CLARIT system to accept the generated indexing set as is. In this way, we have total control over the indexing terms actually used by the CLARIT system, and the CLARIT system essentially serves only as a “blind” retrieval engine, although the thesauri-based feedback mechanism in CLARIT is still used.

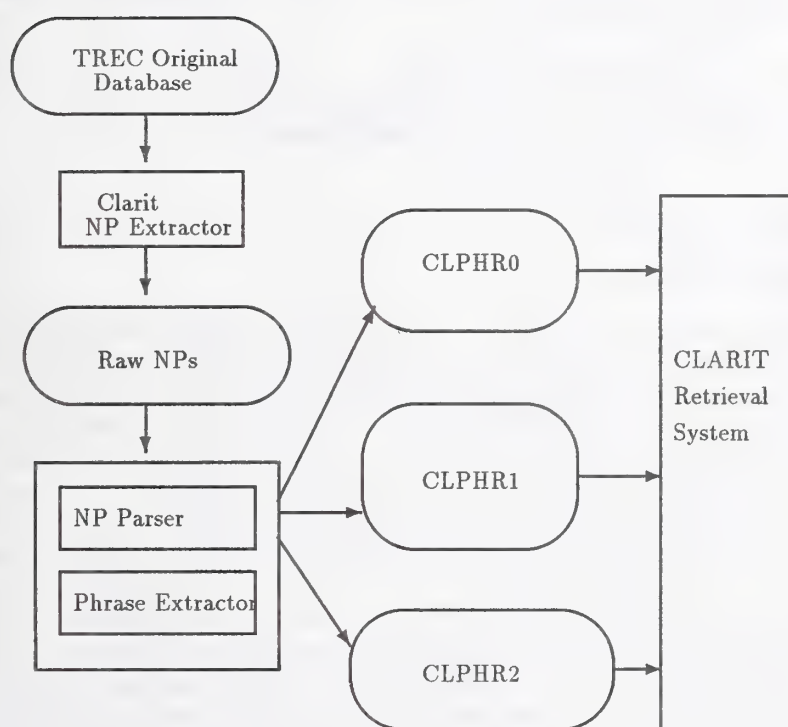


Figure 1: Procedure for Phrase Combination Experiments

The noun phrase parser uses an Expectation Maximization (EM) algorithm to obtain statistical evidence of word modifications from the noun phrases in the corpus. Such evidence is then used to produce an internal modification structure for each noun phrase. The basic idea is as follows. A noun phrase can be assumed to be generated from a word modification structure. Since noun phrases with more than two words are structurally ambiguous, if we only observe the noun phrase, then the actual structure that generates the noun phrase is “hidden”. We therefore treat the noun phrases with their possible structures as the complete data and the noun phrases occurring in the corpus (without the structures) as the observed incomplete data. In the training phase, an EM algorithm can be used to estimate the parameters of word modification probabilities by maximizing the conditional expectation of the likelihood of the complete data (i.e., noun phrases with explicit structures) given the observed incomplete data (i.e., noun phrases in the corpus). In the parsing

phase, a noun phrase is assigned the structure that has the maximum conditional probability given the noun phrase (See [Zhai 97] for more details). Once the structure of a noun phrase is available, four different kinds/levels of terms may be generated from the noun phrase by the phrase extractor: (1) single words; (2) head modifier pairs (i.e., any word pair that has a linguistic modification relation); (3) adjacent subphrases; and (4) the full noun phrase. For example, from the phrase structure "[[[heavy=construction]=industry]]=group]" (a real example from WSJ90), it is possible to generate the following candidate terms:

```
heavy, construction, industry, group (single words)
construction industry, industry group, heavy construction (head modifiers)
heavy construction industry (adjacent subphrase)
heavy construction industry group (full noun phrase)
```

Different combinations of the four kinds of terms can be selected for indexing. In particular, the indexing set formed solely of single words is used as a baseline to test the effect of using phrases. For example, the three automatic official runs (i.e., CLPHR0, CLPHR1, and CLPHR2) correspond to the following three levels of term combinations:

```
-- CLPHR0: single word only (no phrases, baseline)
-- CLPHR1: single word + head modifier pair + full NP
-- CLPHR2: single word + head modifier pair + adjacent subphrase + full NP
```

4 Result analysis

4.1 Lexical atom experiments

In lexical atom experiments we used the same topic profiles that were used in the main manual ad hoc task. The profiles were created manually with the aid of the CLARIT thesaurus extraction module. In addition to the term vector representation of the topic, they also contain Boolean type constraints which may be used to select documents for CLARIT automatic feedback. (See [Milić-Frayling et al. 97] for details of the construction procedure.) The two runs that we submitted for evaluation, CLATMN and CLATMC, differ in the use of constraints. CLATMN topic profiles involve only the query term vector, while CLATMC uses both the query term vector and the constraints. Therefore, automatic query augmentation in CLATMN is based on the top 50 documents in the list of documents retrieved by using the initial query term vector. In CLATMC, however, the list of the retrieved documents is reranked by "pulling up" the documents satisfying the constraints, and the feedback involves the top 50 documents in the reranked list of documents.⁵

In order to include lexical atoms into database index, we first identified approximately 13,000 lexical atoms from the WSJ90 database using the method described in [Evans & Zhai 96], and generated the extended CLARIT lexicon. Both CLATMC and CLATMN used the extended lexicon to parse the document text and query. For the purpose of comparison, we performed the experiments that use the standard CLARIT lexicons: CLATMC-Base and CLATMN-Base, and treat them as the baseline runs. Since the only difference between the official and baseline runs is in their respective lexicons, the difference in the retrieval performance is attributed to the addition of lexical atoms. The results are shown in Figure 2 and Table 1.

From Table 1, we see that indexing with lexical atoms yields higher recall and average precision but a lower initial precision than the corresponding baseline runs. This is different from the results that we obtained in our preliminary experiments using TREC-4 topics. For TREC-4 topics, the lexical atoms improved average precision and initial precision but did not consistently help recall. (See Table 2.)

From both the TREC-5 and the preliminary experiments we see that the use of lexical items leads to a slight but consistent improvement in average precision. This indicates that using lexical atoms to replace single words for indexing generally results in a greater retrieval accuracy than

⁵If the total number of documents that satisfy the constraints is less than 50, it is possible that fewer than 50 documents are actually used for feedback.

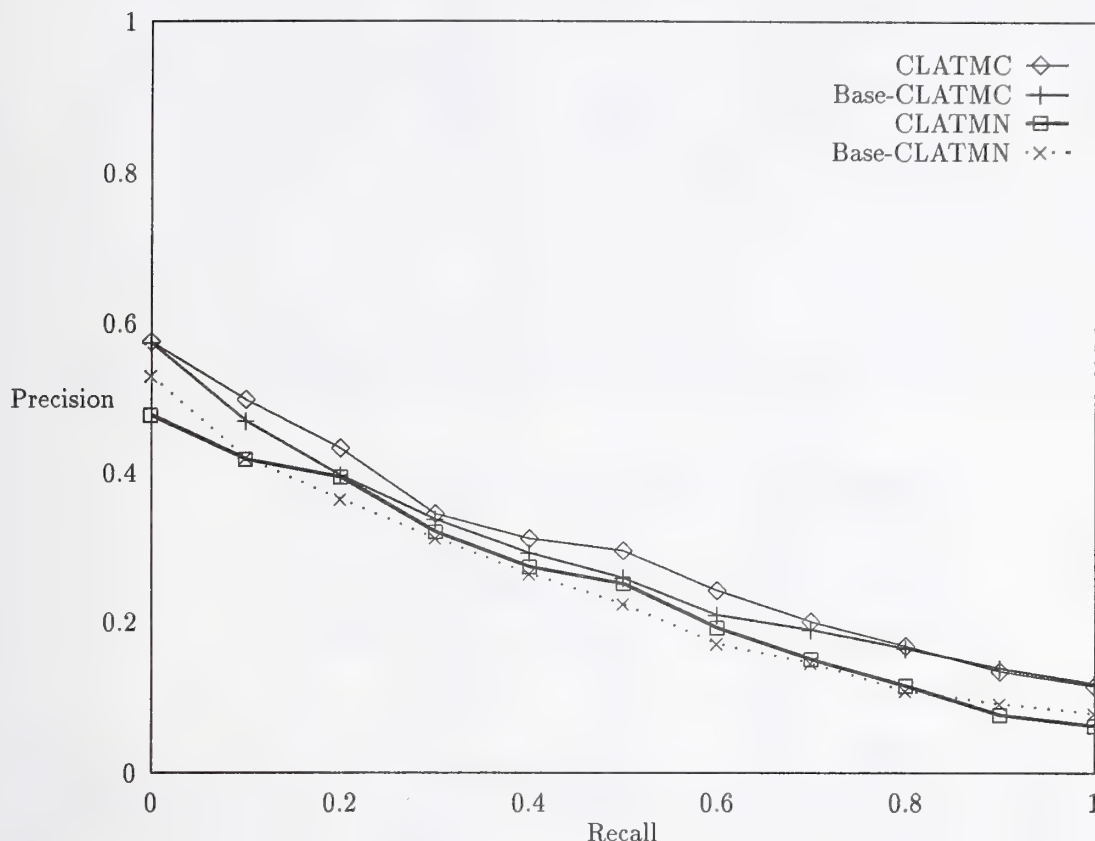


Figure 2: Precision and recall for lexical atom experiments

indexing with single words alone. On the other hand, it seems that using lexical atoms does not consistently improve recall and initial precision. In fact, it seems that it increases either recall or the initial precision.

A brief manual inspection of the identified lexical atoms shows that many, in fact, most, are not true lexical atoms. However, they are usually good terminological phrases such as “stock market” and “annual meeting”. Thus, the results discussed above are in fact the results of using a very rich set of good terminology, including true lexical atoms, to replace the corresponding single words. We believe that the above observed inconsistency in the effect on initial precision and recall may be resolved by a better control over the use of these phrases. (For example, some non-lexical-atoms should supplement rather than replace single words. We should distinguish them from lexical atoms.) We would also like to point out that, in our approach, a single word contained in a lexical atom will still be used for indexing, if the word occurs separately in another non-lexical-atom phrase. We believe that in most cases, lexical atoms only reduced the frequency of the contained single words, and the single words might still be used for indexing owing to other occurrences of the words.

4.2 Phrase combination experiments

In TREC 5, we submitted three official experiments which use different combinations of phrases for indexing: CLPHR0, CLPHR1, and CLPHR2. CLPHR0 used only single words and serves as our

RUNs	Recall(Ret-rel)	Init Prec	Avg Prec
CLATMN-BASE	0.71(753)	0.5293	0.2270
CLATMN	0.74(786)	0.4767	0.2346
increase	4%	-9%	3%
CLATMC-BASE	0.70(748)	0.5744	0.2659
CLATMC	0.72(766)	0.5752	0.2842
increase	3%	0%	7%
Total relevant documents: 1064			

Table 1: Lexical atom experiments

RUNs	Recall(Ret-rel)	Init Prec	Avg Prec
Auto-BASE	0.80(944)	0.4992	0.2291
Auto-LEX	0.76(901)	0.5114	0.2350
increase	-4%	2%	1%
Manual-BASE	0.89(1058)	0.7140	0.3631
Manual-LEX	0.90(1063)	0.7384	0.3701
increase	0%	3%	2%
Total relevant documents: 1183			

Table 2: Lexical atom preliminary experiments on TREC-4 topics

baseline; CLPHR1 used single words, head modifier pairs, and the full noun phrase; CLPHR2 used adjacent subphrases in addition to those terms used by CLPHR1. We also performed the experiment CLPHR0.5 which uses single words and head modifier pairs only. In all these experiments we used the long format of the topic, i.e., all the fields in the topic description. In each experiment, the queries were processed in the same way as the documents. This means that the same combination of phrases is used for both the document and the query. Moreover, the queries were processed with one round of automatic feedback using the top 10 documents from the initial retrieval.⁶

RUNs	Recall(Ret-rel)	Init Prec	Avg Prec
CLPHR0	0.53(567)	0.4235	0.1834
CLPHR0.5	0.57(607)	0.4585	0.2064
inc over CLPHR0	8%	8%	13%
CLPHR1	0.57(609)	0.4465	0.2010
inc over CLPHR0	8%	5%	9%
CLPHR2	0.58(622)	0.4316	0.1997
inc over CLPHR0	9%	2%	9%
Total relevant documents: 1064			

Table 3: Effects of Phrases in the official TREC-5 runs

In Table 3 we show the comparison of the experiments that use phrases with the baseline run that uses only single words. We note that the use of phrases to supplement single words yields between 8% and 9% relative increase in recall and between 9% and 13% relative increase in average precision. Figure 3 in Appendix A provides the per-topic comparison between CLPHR0 and CLPHR1 in terms of their recall and average precision. It is clear from this figure that the general improvement in

⁶Using only 10 documents for feedback yields a very selective (and limited) set of supplemental terminology. There is no guarantee that the additional terms add relevant phrasal combinations.

performance of CLPHR1 over CLHR0 does not come from an even improvement on all topics. In fact, for some topics, the results of CLPHR1 are better, while for others, the results of CLPHR0 are better, and the difference in performance is more significant for some topics than for others.

The relative merits of different phrase sets are unclear. For example, while CLPHR2 produces a higher recall than CLPHR0.5, its precision is lower than for CLPHR0.5. Figure 4 in Appendix A shows the per-topic comparison between CLPHR1 and CLPHR2 in terms of their recall and average precision. Although the picture in this figure looks very similar to that in Figure 3, the scales in Figure 4 are all much smaller than in Figure 3, indicating that the difference between CLPHR1 and CLPHR2 is much less significant than that between CLPHR0 and CLPHR1. Since CLPHR2 uses more phrases than CLPHR1, the mixed relative improvement shown in Figure 4 indicates that adding more phrases does help some topics, but may also hurt some others.

In the official runs, we did not use the frequency of terms in the query for calculating the term weight. In subsequent experiments (CLPHR0-TF, CLPHR0.5-TF, CLPHR1-TF, and CLPHR2-TF), we found that using query term frequency consistently improves both recall and precision, as shown in Table 4.

RUNs	Recall(Ret-rel)	Init Prec	Avg Prec
CLPHR0	0.53(567)	0.4235	0.1834
CLPHR0-TF	0.56(597)	0.4546	0.2208
increase	5%	7%	20%
CLPHR0.5	0.57(607)	0.4585	0.2064
CLPHR0.5-TF	0.60(637)	0.5125	0.2398
increase	5%	12%	16%
CLPHR1	0.57(609)	0.4465	0.2010
CLPHR1-TF	0.63(671)	0.4845	0.2275
increase	10%	9%	13%
CLPHR2	0.58(622)	0.4316	0.1997
CLPHR2-TF	0.62(661)	0.4839	0.2337
increase	6%	12%	17%
Total relevant documents: 1064			

Table 4: Comparison of official and post-TREC-5 runs

The retrieval performance seems to be sensitive to the weighting of query terms. But note that the same effect of using phrases for indexing can be observed when the queries are processed with/without the term frequency, as shown in Table 5.

Our preliminary experiments with TREC-3 topics in the default mode, without using term frequency in the query, also show the positive effect of phrases. (See Table 6.) In fact, the relative improvement in average and initial precision for TREC-3 topics is much higher, between 10% and 23% for the average precision and 17% to 23% for the initial precision. Relative increase in recall is between 5% and 7%. The experiments with TREC-3 queries also show consistent improvement in performance as more phrases are added, PRE-CLPHR2 being the best of the three phrase runs. However, this trend in performance improvement was not observed in TREC-5 experiments.

In Table 5, we also include information about the NLP baseline runs for the SMART system, performed by the Xerox group. XEROX_NLP1 is pure word-based SMART, using SMART stemming; XEROX_NLP2 is pure SMART with SMART-like contiguous phrases. Therefore, XEROX_NLP1 is comparable to CLPHR0-TF; while XEROX_NLP2 is comparable to CLPHR0.5-TF, CLPHR1-TF, and CLPHR2-TF. In comparison with the SMART baseline runs, the CLARIT system generally achieves better precision and worse recall. Indeed, from the performance statistics of the two baseline runs, CLPHR0-TF and XEROX_NLP1, it seems that the selection of single words for indexing by the CLARIT system is more attuned to higher precision than recall.⁷ The phrases that are used

⁷In TREC 5, the CLARIT ad hoc runs and NLP runs both found the largest number of unique relevant documents

RUNs	Recall(Ret-rel)	Init Prec	Avg Prec
CLPHR0-TF	0.56(597)	0.4546	0.2208
CLPHR0.5-TF	0.60(637)	0.5125	0.2398
inc over CLPHR0-TF	7%	13%	9%
CLPHR1-TF	0.63(671)	0.4845	0.2275
inc over CLPHR0-TF	12%	7%	3%
CLPHR2-TF	0.62(661)	0.4839	0.2337
inc over CLPHR0-TF	11%	6%	6%
XEROX_NLP1	0.62(664)	0.4673	0.2002
XEROX_NLP2	0.63(672)	0.4789	0.2147
inc over XEROX_NLP1	1%	2%	7%
Total relevant documents: 1064			

Table 5: Phrase effect on post-TREC-5 runs

RUNs	Recall(Ret-rel)	Init Prec	Avg Prec
PRE-CLPHR0	0.77(1576)	0.6183	0.2782
PRE-CLPHR0.5	0.81(1653)	0.7234	0.3061
inc over PRE-CLPHR0	5%	17%	10%
PRE-CLPHR1	0.82(1664)	0.7263	0.3377
inc over PRE-CLPHR0	6%	17%	21%
PRE-CLPHR2	0.83(1687)	0.7633	0.3416
inc over PRE-CLPHR0	7%	23%	23%
Total relevant documents: 2041			

Table 6: Effects of phrases in the preliminary experiments with TREC-3 topics

by the CLARIT system, on the other hand, lead to a relative increase in average precision that is comparable with the increase for the SMART system but much higher for the recall and the initial precision. Indeed, while phrases used by SMART help improve recall by 1% and the initial precision by 2%, CLARIT phrases result in the relative increase between 7% and 12% for recall and between 6% and 13% for the initial precision.

5 Conclusion

The CLARIT NLP track effort has evaluated the effectiveness of the syntactic phrases generated by a statistical NLP module based on the output from the CLARIT noun phrase extractor. The NLP module is very fast and can be used to perform the experiments on the full set of ad-hoc databases.⁸ Results of our official and post-TREC-5 experiments have shown that the two hypotheses that we proposed in Section 2 are plausible.

Our lexical atom experiments have shown that exploiting lexical atoms to replace or simply reduce the frequency of the single words that form lexical atoms consistently (although only slightly) increases the average precision. The inconsistent influence of lexical atoms on the recall and initial precision may indicate a need for a better control over the selection of phrases that are used for replacing single words.

among all the participating groups. It will be important, subsequently, to investigate the reasons why the CLARIT system found so many documents that other systems missed.

⁸It takes about 1 hour to parse the whole WSJ90 on a 133-MHz DEC Alpha workstation, and the training takes about 5 hours.

Experiments in supplementing single words by various combination of syntactic phrases in the indexing process have shown a consistent and significant improvement in retrieval performance. The general conclusion that syntactic phrases have positive effects when supplementing single words for indexing is also supported by experiments conducted by other groups such as Xerox [Hull et al. 96] and GE/Lockheed Martin/Rutgers/NYU [Strzalkowski et al. 97]. In our experiments, this effect of phrases seems to hold even in combination with other techniques, such as using query term frequency for term weighting. However, the impact of adding phrases into the index space varies according to the query topic. Thus, while adding phrases helps some topics, it hurts some others. Although the results of the performed experiments are revealing, we still need to conduct more experiments to understand how to combine single words and phrases more effectively and to determine the merits of different types of phrases. We also need to analyze varying effects of phrases across different topics.

One thing we need to keep in mind when interpreting these results is that the effect of phrases can be best shown when the phrases also occur in the queries. However, all the results we have discussed came from the experiments using TREC-5 ad-hoc topics, which generally contains very few phrases. The lack of phrases in the queries makes it even more difficult to show the effect of using additional phrases for indexing, since such additional phrases (discovered by the NLP module) have even a smaller chance of occurring in the queries⁹. It is, thus, not a surprise that the GE/Lockheed Martin/Rutgers/NYU group has found that syntactic phrases seem to be more effective with longer queries [Strzalkowski et al. 97]. In fact, the additional phrases that have been used for indexing, but have not shown up in the queries, may have negative side-effects when weighted inappropriately (cf., for example, [Buckley 93]). Thus, phrase weighting is an important factor in the proper evaluation of phrase-based indexing. Based on the above observations, we believe that it would be more appropriate to evaluate phrase-based indexing under the routing task, since, with training documents available, it is possible to formulate queries with more phrases found in the training documents.

References

- [Belkin & Croft 87] Belkin, N., and Croft, B. Retrieval techniques. In: Williams, Martha E.(Ed.), *Annual Review of Information Science Technology*, Vol. 22. Amsterdam, NL: Elsevier Science Publishers. 1987. 110-145.
- [Buckley 93] Buckley, Chris. The importance of proper weighting methods. In: *Human Language Technology. Proceedings of a workshop held at Plainsboro*, New Jersey, March 21-24, 1993. 349-352.
- [Evans et al. 95] David A. Evans, and Robert G. Lefferts. CLARIT-TREC experiments, *Information Processing and Management*, Vol. 31, No. 3, 1995. 385-395.
- [Evans et al. 91] David A. Evans, Kimberly Ginther-Webster, Mary Hart, Robert G. Lefferts, Ira A. Monarch. Automatic indexing using selective NLP and first-order thesauri. In: A. Lichnerowicz (Ed.), *Intelligent Text and Image Handling. Proceedings of a Conference, RIAO '91*. Amsterdam, NL: Elsevier. 1991. 624-644.
- [Evans & Zhai 96] Evans, D. and Zhai, C. Noun-phrase analysis in unrestricted text for information retrieval. *Proceedings of the 34th Annual meeting of Association for Computational Linguistics*, Santa Cruz, University of California, June 24-28, 1996. 17-24.
- [Fagan 87] Fagan, Joel L. *Experiments in Automatic Phrase Indexing for Document Retrieval: A Comparison of Syntactic and Non-syntactic Methods*. PhD thesis, Dept. of Computer Science, Cornell University. Sept. 1987.
- [Hull et al. 96] Hull, D. et al. Xerox TREC-5 Site Report: Routing, Filtering, NLP, and Spanish Tracks, In: *TREC-5 Conference Notes*. 1996.

⁹However, our experiments with TREC-3 topics did show a consistent improvement of performance when more phrases are used for indexing.

- [Lewis 91] Lewis, D. *Representation and Learning in Information Retrieval*. Ph.D thesis, COINS Technical Report 91-93, Univ. of Massachusetts. 1991.
- [Milić-Frayling et al. 97] Milić-Frayling, N. Tong, X., Zhai, C., and Evans, D. A. CLARIT Compound Queries and Constraint-controlled Feedback in TREC-5 Ad Hoc Experiments. In: D. Harman (Ed.) *The Fifth Text REtrieval Conference (TREC-5)*. NIST Special Publication, 1997. (This Volume.)
- [Strzalkowski et al. 95] Strzalkowski, T., et al. Natural language information retrieval: TREC-3 report. In: Harman, D. (Ed.), *Overview of the Third Text REtrieval Conference (TREC-3)*. NIST Special Publication, 1995. 500-225.
- [Strzalkowski et al. 97] Strzalkowski, T., et al. Natural Language Information Retrieval: TREC-5 Report. In: D. Harman (Ed.), *The Fifth Text REtrieval Conference (TREC-5)*. NIST Special Publication, 1997, forthcoming.
- [Zhai 97] Zhai, Chengxiang. Fast statistical parsing of noun phrases for document indexing. To appear in *Proceedings of the Fifth Conference on Applied Natural Language Processing*, Washington, D.C. March 31 – April 3, 1997.

A Per-topic comparisons

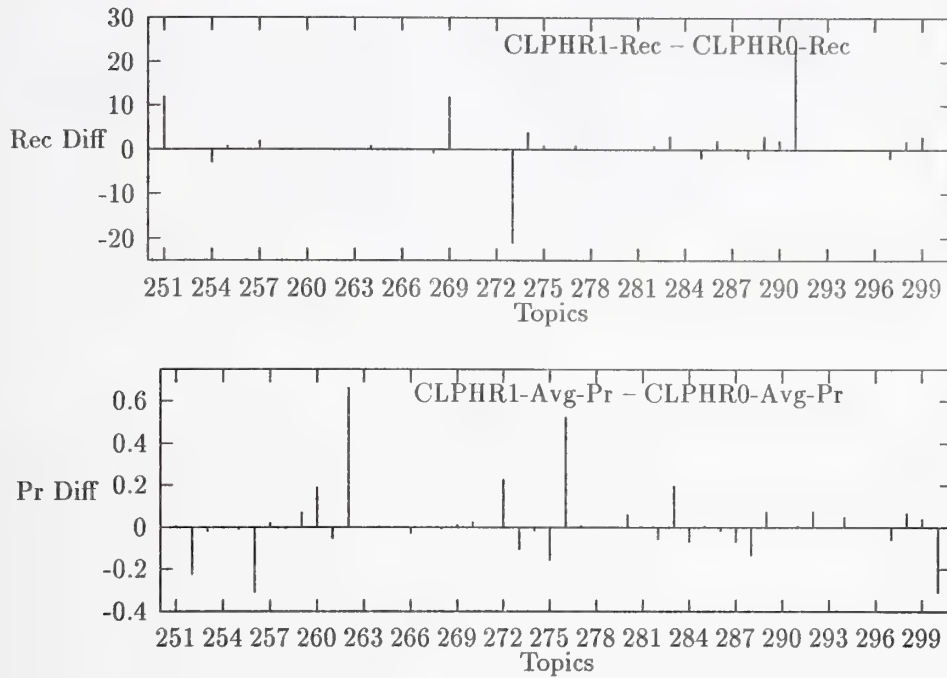


Figure 3: Per-topic Comparison of CLPHR0 and CLPHR1

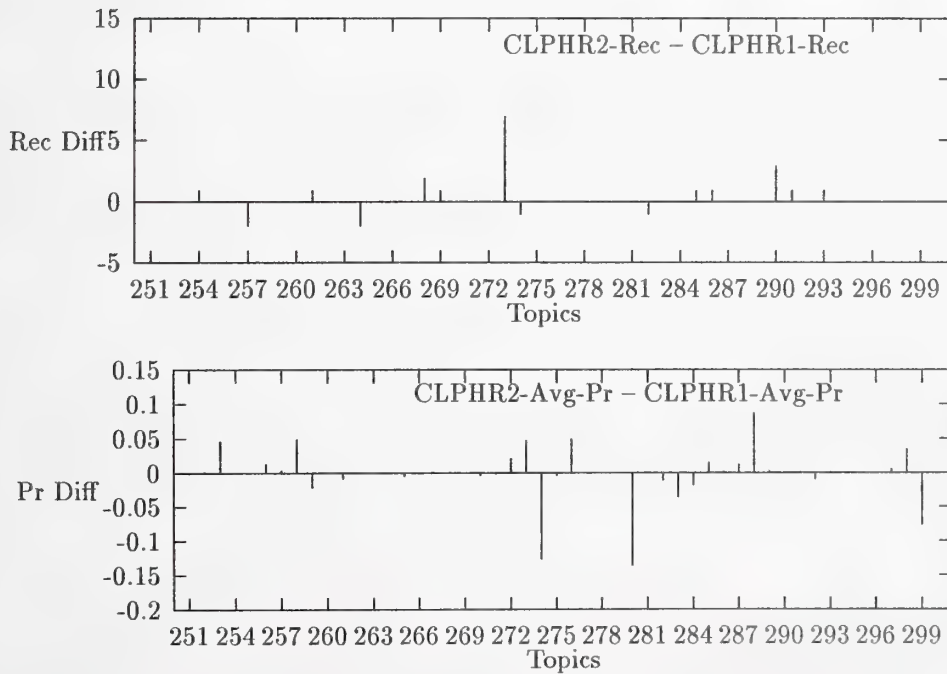


Figure 4: Per-topic Comparison of CLPHR1 and CLPHR2



ANU/ACSys TREC-5 Experiments

David Hawking, Paul Thistlewaite and Peter Bailey
Co-operative Research Centre For Advanced Computational Systems
Department Of Computer Science
Australian National University
{dave,pbt,peterb}@cs.anu.edu.au

February 1, 1997

Abstract

A number of experiments conducted within the framework of the TREC-5 conference and using the Parallel Document Retrieval Engine (PADRE) are reported. Several of the experiments involve the use of distance-based relevance scoring (spans). This scoring method is shown to be capable of very good precision-recall performance, provided that good queries can be generated. Semi-automatic methods for refining manually-generated span queries are described and evaluated in the context of the adhoc retrieval task. Span queries are also applied to processing a larger (4.5 gigabyte) collection, to retrieval over OCR-corrupted data and to a database merging task. Lightweight probe queries are shown to be an effective method for identifying promising information servers in the context of the latter task. New techniques for automatically generating more conventional weighted-term queries from short topic descriptions have also been devised and are evaluated.

1 Introduction

The work reported here comprises a number of text retrieval experiments conducted within the framework of the TREC-5 task and addressing questions of interest in the following research areas: Applications of parallelism in information retrieval; Distance-based relevance scoring; Distributed IR; and Automatic query generation.

In TREC-5, ANU/ACSys runs were submitted in Automatic Adhoc, Manual Adhoc, DB Merging, Confusion and VLC categories.

1.1 Hardware and Software Employed

A 128-node, distributed-memory Fujitsu AP1000 (16 MBytes of RAM per node, 2 gbytes in total) running PADRE [6] software was used for all retrieval runs reported here. Most runs used the Super Dictionary (SD) method (using disk-resident inverted files) [1] rather than the Full Text Scanning (FTS) method used in previous ANU TREC submissions.

1.2 Statistical Testing of Differences Between Runs

Throughout this paper, many comparisons are made between pairs of runs. In making these comparisons, apparent differences between means have been tested for statistical significance using the well-known *t*-test. Two-tailed tests with a 5% confidence level were used.

Table 1: Comparison of ANU TREC-5 Automatic Adhoc runs with that of TREC-4.

	padreA(TREC-4)	anu5aut1	anu5aut2
Recall	35%	34%	36%
Average Precision	.1453	.1537	.1538
Exact Precision	.1954	.1948	.1980
Precision at 10 docs	.2949	.2540	.2280
Precision at 20 docs	.2857	.2160	.2100
Precision at Recall .10	.3186	.2869	.2898
Precision at Recall .20	.2521	.2399	.2378
Precision at Recall .40	.1722	.1760	.1778
Precision at Recall .50	.1300	.1557	.1575
No. "best" on ave prec	-	6	3
No. "best" on recall	-	8	6

2 Automatic Query Generation

Automatic AdHoc, Official Runs anu5aut1 and anu5aut2

The main activities in ANU/ACSys participation in the Automatic AdHoc category were directed toward:

1. Replacing the semantics-based techniques used in the TREC-4 runs for determining phrases and for selecting and weighting query terms, with techniques based on term frequency alone.
2. Attempting to determine from a topic whether to restrict documents to being USA-only, foreign-only, etc.
3. Developing and evaluating a new stemming algorithm.

Only the short form of the topic descriptions was used.

2.1 Relevance Scoring Method in Automatically Generated Queries

The relevance ranking algorithm was the same as that used by ANU in the TREC-4 Automatic Adhoc task, but with a different term weighting scheme. The terms used for a given topic were determined by extracting all non-function content words and then searching for all co-occurrences of these terms (within varying lexical proximities) to determine suitable multi-word terms. The weights for these phrasal terms were computed as a function of their occurrence frequencies and their expected frequencies.

In the anu5aut2 run, all multi-word terms that were derived were used; in the anu5aut1 run there was a threshold value used to select a subset of the multi-word terms.

All non-function single words from the topic also received a weight proportional to the *idf* for the word.

Results: Table 1 compares the performances of TREC-4 and TREC-5 automatically generated queries averaged across all topics. There was no significant difference

between the two TREC-5 runs on average precision ($t(49) = 0.059$) but `anu5aut2` performed 9% better on average topic-by-topic percentage recall ($t(49) = 2.589$, $p < 0.05$).

Discussion and Conclusions: Comparison of the two TREC-5 runs suggests that attempting to select a subset of the multi-word terms may be counter-productive. The similarity of TREC-5 to TREC-4 results was a source of initial disappointment until it was realised that other groups found the task significantly harder than its equivalent in TREC-4. This observation is corroborated by the big improvement in the number of “best” scores achieved (see table 1). Although early precision worsened relative to TREC-4, average precision improved slightly due to improved precision in middle to later stages (0.4 recall level onwards). The fact that average precision results were similar to those of leading groups prior to query expansion indicates that the current method will provide a promising basis for application of expansion techniques.

3 Effectiveness of Distance-Based Relevance Scoring

The query set employed in ANU’s TREC-4 Manual Adhoc submission in TREC-4, (which will be referred to here as Q_{T4}^{ANU}) scored document relevance using only the lexical distance between instances of concepts in concept intersections (*Z-mode*). Clarke, Cormack and Burkowski [3] independently developed a very similar system of distance-based scoring and used it in the University of Waterloo TREC-4 submission. Buckley, Singal and Mitra [2] also used distance-based measures in their Individual Term Locality run `Cn1AL`.

Subsequent to TREC-4, Hawking and Thistlewaite [9] proposed an extended formal model of *spans*. Much of this model has now been implemented in PADRE and was used as the scoring method in the Manual Adhoc, DB Merging, Confusion and VLC runs reported below.

Although distance-based queries performed well in the Manual Adhoc and DB Merging categories of TREC-4, there remained a gap between their performance and that of the best conventionally-ranked queries. The goal of the present experiment was to determine whether the gap represented a fundamental limitation of distance-based scoring or whether the problem lay in the quality of the queries. A partial answer was sought by attempting to create a set of high-standard distance-based queries and to compare its performance with that of the best TREC-4 systems. The new query set was constructed by selecting the best-performing of three independently generated queries for each topic. Each query was constructed without reference to retrieved documents.

The three query sets used comprised Q_{T4}^{ANU} , the official Waterloo TREC-4 queries (Q_{T4}^{UW}), and a new set of span-based TREC-4 queries (Q_{T4}^{T5prac}) generated to practice a new manual approach to query generation. The author of the latter queries was the same person who generated the Q_{T4}^{ANU} set but it was hoped that a period of two months without exposure to the topics or documents would permit the second set to be relatively independent of the first. These query sets achieved average precision results of 0.2383 (Q_{T4}^{ANU}), 0.2994 (Q_{T4}^{UW}) and 0.2898 (Q_{T4}^{T5prac}), compared with 0.3436 for the overall best official TREC-4 run (`CnQst2`, submitted by Excalibur Technologies Inc).

The sixteen queries from the Q_{T4}^{UW} set which performed more than 0.100 better on average precision than the corresponding ones from Q_{T4}^{ANU} were translated into PADRE format. It was verified that each translated query achieved similar average precision to the Waterloo origi-

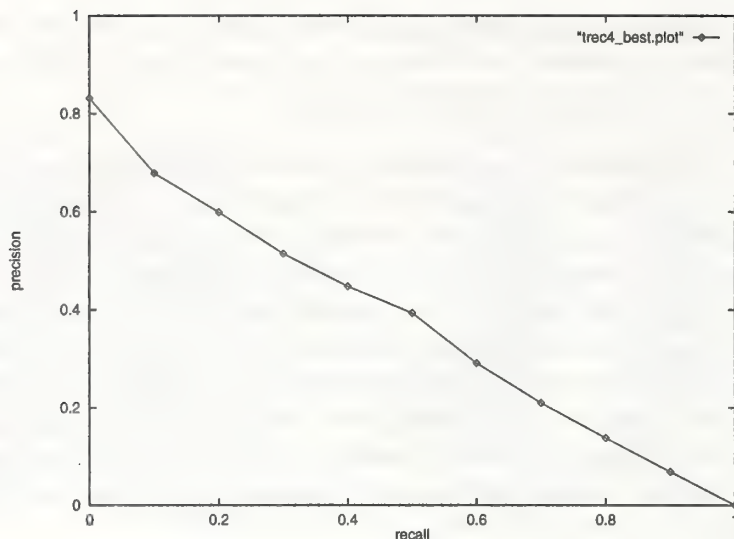


Figure 1: Precision-recall curve for the best available PADRE queries for the TREC-4 task. They were scored entirely according to span distances. The distance decay function employed was the custom function shown in figure 3. The maximum score achievable by a partial span was reduced by a factor of 10 for every term missing. A proximity limit of 1000 characters applied to spans.

nal. The translated queries then replaced the corresponding inferior versions in Q_{T4}^{ANU} to form $Q_{T4}^{ANU/UW}$. This query set achieved an average precision of 0.3208.

Finally, by converting the $Q_{T4}^{ANU/UW}$ queries to PADRE's new span formulation (which scores partial spans appropriately) and merging the best of these queries with the best of Q_{T4}^{T5prac} , the set Q_{T4}^{best} was formed.

Results: The Q_{T4}^{best} queries retrieved 69.7% of all relevant documents, averaged 6.3 relevant documents in the first 10 and achieved an average precision of **0.3634**. As shown in the topic-by-topic results for this query set (appendix A), average precision results are better than the median of all runs for 44 of the 49 topics.

Discussion and Conclusions: These results compare very favourably with corresponding figures of 71.0%, 5.7 and 0.3436 for the best official TREC-4 run (CnQst2). As it is not yet certain whether distance-based queries are suitable for all topics, it would be useful to further study the queries for the five below-median topics with a view to determining whether any of them appear to be intractable to span scoring. Although the process of selecting the best of different sets of queries violates TREC-4 rules, it is justifiable to conclude that use of distance-based scoring alone does not limit performance to a level below that of state-of-the-art, conventionally scored systems. However, methods for reliable, low cost generation of high-standard distance-based queries are clearly needed.

4 Manual (Non-Interactive) Query Generation

Manual AdHoc, Official Runs anu5man4 and anu5man6

The following goals motivated the work in the Manual AdHoc category:

```

topic 251
weight 0
anyof "exportation |exporting |offshore "
anyof "factories |factory |industries |industry |manufactur"
loadmatchset USall
loadmatchset countries
anyof "employ|job |jobs |unemploy"
span key 5 1000 2000 2 2
top 1000

```

Figure 2: The Q_0 query for the topic relating to the exportation of industry. USall and countries are pre-computed matchsets resulting from searches for very large numbers of words, phrases and abbreviations indicating the USA and other countries respectively. The span command scores the relevance of documents containing spans across all 5 matchsets, of which the first two are mandatory. Documents containing instances of all five matchsets within a proximity of 2,000 characters receive a relevance score contribution depending upon the length of the span. Partial spans, for example those missing an indication of a foreign country, will attract a lower score contribution than would a complete span of the same length.

1. To confirm or otherwise the precision-recall effectiveness of distance-based measures alone,
2. To reduce the amount of human time spent in writing queries,
3. To reduce reliance on subject experts during query construction, and
4. To explore automatic query construction aids for span queries.

4.1 Query Structure

Manual Adhoc queries were constructed using concept intersections. Queries consisted of 1 - 3 spans. Spans contained 1 - 5 branches. The enhanced span model allows scoring of partial spans and also allows singleton spans, which differ from conventional terms only in that *idf* plays no part whatever. In certain topics, such as **self induced hypnosis** and **treatment of schizophrenia** from TREC-4, the use of singleton spans is critical to good ranking.

4.2 Overview of Manual Query Generation Process

In order to study the effectiveness of various query enhancement techniques, an initial set of queries was subjected to successive refinements and each resulting set was run against the TREC-5 data. The initial set of queries was drafted without using any information from the collection. Refinements used term-term implication and partial-match frequency information derived from the collection. Finally, the refined queries for some topics were augmented with manually generated regular expressions for numbers, dates and currency amounts.

A utility program **qgreen** was developed and used to check queries for errors, inconsistencies and inefficiencies. No retrieved documents were examined manually at any stage in manual query construction.

4.3 Construction of Initial (Draft) Queries

Queries were again generated by the first author. Last year, considerable time was invested in constructing the queries and considerable help was obtained from subject experts. In some cases this resulted in unnecessarily elaborate queries constructed around terms which are important in the subject as a whole but which appear too infrequently in the collection to be useful.

This year, in line with the goals stated above, no experts were consulted and only about half the amount of time was allowed for the manual input. (See table 2.)

The set of initial queries thus constructed was called Q_0 . It was used in [unofficial] run `anu5man1`. An example query is shown in figure 2.

Results: The results achieved by `anu5man1` were considerably worse than the corresponding `padre2` run from TREC-4. Average precision dropped from 0.2383 to 0.1973. Early precision and overall recall also declined markedly.

Discussion and Conclusions: The decline in performance relative to the same category in TREC-4 is partly due to the increased task difficulty. No doubt there is also a relative decline due to reduced query input time and lack of subject-expert advice. In this context it is difficult to estimate whether any benefit was conferred by the increased sophistication of span scoring.

After construction of the initial query for each topic a number of partial queries were extracted, to be used in term implication runs explained below. The number of partial queries used for a topic ranged from one to five.

4.4 Query Augmentation

Robertson [10] argues that the best methods for selecting terms for query expansion and for weighting selected terms are not necessarily the same. In distance-based queries, as formulated here, this difficulty is avoided because individual terms are not weighted. However, a new complication is introduced.

When augmenting distance-based queries using concept intersections, it is necessary not only to find new terms but also to identify which concept they augment. New terms found to be strongly associated with documents scoring highly against a query may be good candidates for addition. However, if a particular new term is added to the definition of the wrong concept, then certain documents will score artificially highly. For example, if the term BFI (a well-known recycling company) were added to the `tire` concept rather than the `recycling` concept in a 3-way intersection addressing the economic impact of recycling tires then some totally `tire-free` documents might achieve undeservedly high scores.

In term association, terms are sought which tend to co-occur with a query term (which might be a stem, a word, a phrase or a complex sub-query). All documents matching the query term and all those containing candidates for association are considered. The strength of association between a candidate term t_c and the query term t_q can be expressed as:

$$A_{cq} = \frac{|D_c \cap D_q|}{|D_c \cup D_q|}$$

where D_c and D_q are the sets of documents containing t_c and t_q respectively.

In the experiments reported here, implication strengths rather than association strengths were used. These are directional.

$$I_{cq} = \frac{|D_c \cap D_q|}{|D_c|}$$

$$I_{qc} = \frac{|D_c \cap D_q|}{|D_q|}$$

I_{cq} measures the extent to which the presence of t_c in a document implies that t_q will also be present. $I_{cq} = 1.0$ iff every document containing t_c also contains t_q , but there may be documents containing t_q which do not contain t_c .

The implication strength machinery was also used to achieve the effect of relevance feedback. The same formulae for implication strength were used, except that D_q became the set of top-ranked documents relative to a query q rather than the complete set of matching documents.

In searching for useful additional terms with which to augment the manual queries two distinct processes were used. These are best described by illustration.

Imagine a draft query consisting of a three-way intersection of concepts such as **economic impact**, **recycling** and **tires**. The relevance feedback process finds the top n (say 10) documents against the partial query and looks for terms which occur in a high proportion of them. The second process uses term association to try to find terms which might be associated with individual concepts, with individual concepts or with sub-groupings of concepts. In the term association process, implication strengths are computed for all collection terms passing low-cut and high-cut frequency filters against a series of partial queries such as: **profit***, **recycl***, **tires**, **recycl* near tires**, **profit* near recycl***.

4.5 First Refinement of Draft Queries. $Q_0 \rightarrow Q_{RF1}$

The set of queries augmented using the results of term implication results over the partial queries ONLY is called Q_{RF1} . It was used in [unofficial] run **anu5man2**.

Results: Run **anu5man2** performed significantly better than **anu5man1** on topic-by-topic percentage recall ($t(49) = 2.448$, $p < 0.05$, observed difference +11%). There was no significant difference in average precision ($t(49) = 1.585$, observed difference +10%) or precision @20 ($t(49) = 2.172$, observed difference +7%).

4.6 Second Refinement of Draft Queries. $Q_{RF1} \rightarrow Q_{RF2}$

Full queries from Q_{RF1} were used in a further term implication run resulting in a set of queries called Q_{RF2} which were used in runs **anu5man3** and **anu5man4**.

Only the following information was used in refining Q_{RF1} :

- Terms implying or implied by the top 10 ranked documents selected by the partial or draft query,
- Terms implying or implied by all documents selected by the partial or draft query,
- The count of documents selected by partial or draft queries.

Table 2: Human time consumed in manual query construction. The regular expressions incorporated in Q_{RF2RX} tended to be repeated in multiple topics and the time quoted is total time spent divided by the number of queries (12) which were augmented with regular expressions. The time to devise (and test) complex one-off regular expressions would be greater than the quoted amount. The quoted amount would have been much less if all 50 queries had required recognition of numbers, dates etc.

Activity	Time per query (min.)
Initial composition	15
Checking	5
$Q_0 \rightarrow Q_{RF1}$	10
$Q_{RF1} \rightarrow Q_{RF2}$	5
$Q_{RF2} \rightarrow Q_{RF2RX}$	3

Results: Apparent differences between means for runs `anu5man4` and `anu5man2` were small (4% in average precision, -2% in precision @20 and 1% in recall) and not statistically significant ($t(49) = 1.334, 0.649, 0.428$ respectively).

Compared to the baseline `anu5man1`, `anu5man4` was significantly better on topic-by-topic percentage recall ($t(49) = 2.264, p < 0.05$, observed difference 12%). Apparent improvements of 5% in precision @20 and 15% were not statistically significant. ($t(49) = 1.117$ and 1.973 respectively.)

Compared to manual runs submitted by other groups, `anu5man4` achieved best or equal best results on seven topics in recall and on three in average precision. However, the total number of relevant documents for many of these topics was quite small (in two cases, as few as one). Run `anu5man4` performed better than or equal to the median on 29 topics for recall and 30 topics for average precision.

Discussion and Conclusions: On the basis of training results using TREC-4 data and judgments, results in the 0.28 - 0.32 average precision range were expected. Actual performance fell far short of this. It may be that expectations based on training with TREC-4 data were unrealistically high due to an overtraining effect. However, it is clear that the TREC-5 task was harder than that of TREC-4.

Overall, run `anu5man4` is understood to have achieved very close to the best results for non-interactive manual submissions. Nearly all participants who achieved better results used queries modified after examination of retrieved documents or contexts. Compared to all manual runs submitted by other groups, `anu5man4` achieved best or equal best results on seven topics in recall and on three in average precision. However, the number of *best* recall results may not be very meaningful as the total number of relevant documents for many of these topics was quite small (in two cases, as few as one). Run `anu5man4` performed better than or equal to the overall median on 29 topics for recall and 30 topics for average precision.

4.7 Numbers, Dates, Percentages and Currency Sums. $Q_{RF2} \rightarrow Q_{RF2RX}$

Twelve topics (numbers 257, 260, 264, 268, 272, 277, 285, 286, 291, 293, 298, and 300) were selected as those most likely to benefit from the recognition of numbers, dates, percentages, and currency amounts. Three of these (264, 293, and 298) requested the imposition of a date filter such as "after 1900".

Table 3: Separation values (equivalent number of words) accorded to intervening document features. A sample of documents was read to make a list of SGML tags which indicate no greater semantic or structural barrier than a sentence break. Twelve such marker pairs were identified. They are the so-called *lightweight* tags. All other SGML tags were assumed to be *heavyweight*.

Feature	Distance (equiv. no. words)
Sentence break	4
Uncertain paragraph break	6
Certain paragraph break	8
Lightweight SGML tag	4
Heavyweight SGML tag	200

It was decided to modify the Q_{RF2} queries for these topics by including regular expressions to achieve the desired effect. This produced a new query set Q_{RF2RX} . The 12 modified queries were then run using PADRE's FTS (Full Text Scanning) method as regular expressions are not supported in SD method.

It was realised before running that the presence of numeric quantities in SGML fields (such as DOCIDs, DATEs, DOCNOs etc) accompanying nearly every document would cause a large number of spurious matches on dates and numbers (but probably not percentages or currency amounts). Two alternative solutions to this problem were identified:

1. Use DTDs for the documents to prevent the regexp search code from looking in these confusing places, and
2. Adapt the span scoring code to significantly increase the span length when an SGML tag was encountered.

Of these, the second method was chosen because it was consistent with a desire to investigate measurement of span length taking into account intervening text features such as sentence breaks, paragraph breaks and SGML tags of various types. (See Hawking and Thistlewaite, 1996[9], p. 13.)

The Q_{RF2} queries were run again using a non-linear scoring algorithm whose salient properties are summarised in table 3 to serve as a baseline for judging the benefit of using regular expressions. Unfortunately, the non-linear span-length algorithm has only been implemented in FTS method and consequently there was a very considerable increase in runtimes. Indeed, processing the regexp version of query 277 took more time than all 50 queries in `anu5man1`!

Results: The introduction of non-linear scoring (coupled with the change from super dictionary method to full text scanning) caused an unexpected decline in performance. Runs `anu5man4` and `anu5man5` used identical queries but relative to the former, `anu5man5` was significantly worse on average precision ($t(49) = 3.131, p < 0.05$, observed difference -18%) and on topic-by-topic percentage recall ($t(49) = 4.401, p < 0.05$, observed difference -16%). An apparent difference of -7% on precision @20 was not statistically significant.

Official run `anu5man6` was evaluated using `anu5man5` as a baseline in order to judge the effectiveness of the regular expressions. Considering only the twelve affected topics, the use of regular expressions made a statistically significant difference to

Table 4: Summary of Manual AdHoc runs.

Run-id	Queries	Method	Scoring	Prox. Lim.	Runtime	Recall	@20	Ave. Prec.
anu5man1	Q_0	SD	1/sqrt	2000	23:35	0.4642	0.3650	0.1973
anu5man2	Q_{RF1}	SD	1/sqrt	2000	28:18	0.5163	0.3910	0.2170
anu5man3	Q_{RF2}	SD	1/sqrt	2000	30:00	0.5203	0.3830	0.2161
anu5man4	Q_{RF2}	SD	custom	1000	32:22	0.5230	0.3840	0.2261
anu5man5	Q_{RF2}	FTS	custom/nls	1000	2:23:34	0.4213	0.3560	0.1849
anu5man6	Q_{RF2RX}	FTS	custom/nls	1000	4:02:59	0.4354	0.3580	0.1889

topic-by-topic percentage recall ($t(11) = 2.554, p < 0.05$, observed difference +13%). Apparent improvements of 2% in precision @20 and 11% in average precision were not significant ($t(11) = 0.261$ and 1.443 respectively).

On a topic-by-topic basis, recall apparently improved on nine topics and deteriorated on only one. Average precision improved on seven and deteriorated on five. Precision @20 improved on six and deteriorated on four.

Discussion and Conclusions: The cause of the decline in performance from run anu5man4 to run anu5man5 is of some concern. Further work is needed to ascertain whether the problem lies with the assignment of separation values, or to an unexpectedly significant difference between SD and FTS methods or to the presence of a bug. Despite this outstanding question, it does appear that regular expressions have contributed something of an improvement, albeit at very great computational cost.

4.8 Relevance Scoring

Some training runs and some unofficial TREC-5 runs (see table 4) investigated the effect of different parameters in the distance-based relevance scoring model.

A number of different functions have been used to estimate the declining weight of relevance evidence as span length increases. Inverse square-root and a custom function (figure 3) were used in runs reported here.

A cut-off proximity limit beyond which the probability of relevance is forced to zero is also applied for efficiency. Its value has also been found to have an effect on precision-recall. If set too high or too low recall and average precision are adversely affected. As shown in table 4, two different values were used in TREC-5 runs.

5 Simulated Server Selection and Result Merging

DB Merging Track, Official Runs anu5mrg0, 1 and 7

Full details of experimentation in the context of the database merging task are given elsewhere [7]. The merging task was interpreted as a simulation of a network server selection problem. In this model, each of the 98 sub-sub-collections was served by a distinct network server, simulated by a processing node on ANU's Fujitsu AP1000 (leaving 30 unused nodes). The Q_0 query set was used to avoid using collection information in query generation.

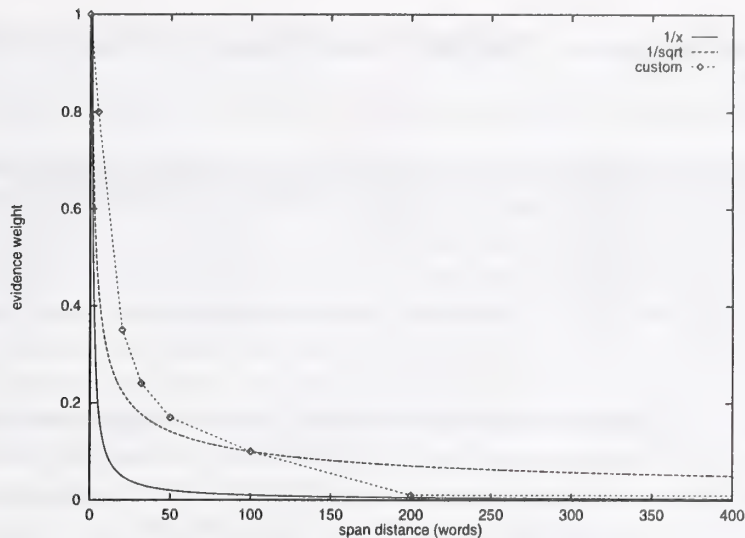


Figure 3: Some possible relationships between strength of evidence represented by a span and its length.

Table 5: Summary of DB Merging runs. All runs used manually-generated query set Q_0 . Percentages in columns 4-6 are relative to the baseline run. Columns 4 and 5 refer to the `trec_eval` measures using the official relevance judgments. Column 6 records the percentage of documents retrieved by the baseline which were also retrieved by the run in question. Figures in parentheses have been scaled-up by $\frac{32}{30}$ as a first-order compensation for the difference in number of sub-sub-collections used.

Run-id	Synopsis of Method	No. Servers	% rel_ret	% ave_prec	% ret
anu5mrg0	baseline	98	100	100	100
anu5mrg1	Topic Similarity	32	63	73	53
anu5mrg7	Lightweight Probes	30	59(63)	61(65)	47(50)

As in TREC-4, distance-based relevance measures only were used. Consequently, the problem of merging rankings from different sub-collections is a non-issue as the distance measures are independent of collection statistics. Results of runs `anu5mrg0` and `anu5man1` are close to identical, as they should be, [8] despite the fact that the former used the database merging division of the collection.

Three runs constitute ANU’s submissions in the DB Merging track. These are summarised in table 5.

The first method (`anu5mrg1`) used TREC-4 topics and queries processed over the entire collection of servers as historical data and sent TREC-5 queries to servers which had proven useful in processing TREC-4 topics which were manually judged to lie in similar subject areas. Similarity judgments are given in appendix A.

The second method (`anu5mrg7`) used no historical information but instead sent light-weight (two-term) probe queries to all servers and used a small packet of frequency information returned in response to select a subset of servers to process the full query.

The goal was to retrieve as high a fraction of relevant documents as possible using only a small subset (about one third) of the available servers.

Results: No significant difference was observed between the methods. Roughly

Table 6: Unofficial Confusion Results. The highest ranked document has rank $R = 0$. Items not found are assigned $R = 1000$.

Run-id	Queries	Collection	No. found	Ranked 0	Ranked < 20	Ave. rank
anu5con0	Q_{con0}	Truth	47	14	35	84
anu5con3	Q_{con0}	Degrade5	44	9	23	197
anu5con1	Q_{con1}	Degrade5	47	12	30	109

speaking, each method retrieved nearly two thirds of the relevant documents retrieved by the baseline, while accessing only about one third of the servers. Note that, in retrieving two thirds of the baseline relevant documents, it was only necessary to retrieve half of all the documents retrieved by the baseline run.

Discussion and Conclusions: Both server selection methods appear to be capable of biasing server selection toward servers which supply documents which achieve non-zero scores and of even more strongly biasing towards those which supply actually relevant documents. The good performance of the lightweight probe method is encouraging as historical query-processing data required by other methods is not always available.

6 Information Retrieval Over OCR-Corrupted Data

Confusion Track, Official Runs anu5con0 and anu5con1

A set of queries Q_{con0} were manually devised using an average of only 3 minutes of human time per topic. They were again based entirely on span scoring. These queries were processed against the “truth” version (anu5con0) and the “degrade5” version of the data (anu5con1).

Full detail of the the method employed is documented in [5]. In summary, characteristic scanning errors were identified in a small sample of the text by comparing truth and degrade5 versions. All combinations of presence/absence of these characteristic errors in each term were then applied to the Q_{con0} queries by a preprocessor, resulting in significantly longer queries Q_{con1} .

Results: Using official expected run-lengths, ANU’s baseline run was 65% worse than the median and the degrade5 run was 48% worse. Relative to the best in each category, the corresponding figures are 1,684% worse and 589% worse. Expected run-lengths were 65% of the worst on the baseline and 22% of the worst on degrade5.

Discussion and Conclusions: It is tempting to conclude that the 3-minute ANU queries were very poor but that the method for compensating for OCR errors was relatively effective. It might be considerably more so if a more systematic approach were taken to identifying the characteristic errors.

7 Experiments with a Larger Collection

Very Large Collection Track, Official Runs `anu5vlc2` and `anu5vlc3`

The collection used in the VLC pre-track comprised all four TREC CDs, a total in excess of 4 gigabytes. This factor of 2 increase over the TREC mainstream task is potentially significant as it goes beyond the amount of data which can be addressed using 32 bits.

In the hope of boosting early precision, VLC runs used the inverse square root of distance as the decay function rather than the custom function and imposed a much more severe penalty on scores derived from partial spans. (In the VLC runs the shortest partial span involving $k - 1$ terms would score just less than the longest admissible span involving k terms. In the manual run, a partial span involving $k - 1$ terms would score 0.1 of the score for a k term span of the same length.)

The baseline run was carried out over the data as organised for the DB Merging run, a super-dictionary linking 5 distributed index files, and only 98 processing nodes were used. The VLC run proper used 9 distributed indexes, the five from DB Merging and two for each of CD1 and CD3.

Table 7: Summary of VLC measures for the ANU submissions.

Measure	<code>anu5vlc2</code> Baseline	<code>anu5vlc3</code> VLC	VLC/Baseline
Precision@20	0.3920	0.5020	1.28
Query processing time	44.8	68.5	1.53
Data struct. bld. time	2177	4721	2.169
Disk space	6.29 gB	10.29 gB	1.63
Memory			

Results: Table 7 shows the VLC measures taken from the two ANU runs in the trial VLC track. Both runs used the Q_{RF2} queries. The baseline run for the VLC task is thus comparable to the `anu5man4` manual adhoc run except for a different model of span scoring. It is not clear how to report memory use on the parallel machine. Since the current operating system enforces a single-user mode of operation, one could say that memory use for both methods is 2 gigabytes, even though that includes space for kernels, PADRE executables, message buffers and unused freespace, all replicated 128 times.

Using NIST assessments, the VLC baseline run scored 0.3920 on precision@20 compared with 0.3840 for `anu5man4`.

Discussion and Conclusions: Early precision was significantly better on the 4 gigabyte collection than on the 2 gigabyte collection. ($t(49) = 3.243, p < 0.05$, observed difference +28%.) A similar phenomenon was observed by other participants in the task. Unfortunately, because the two sets of results were judged by different assessors, difference between judges cannot be ruled out as an explanation. However, it may be that a small class of relevant documents contains so many obvious relevance features that its members will appear ahead of [nearly] all irrelevant documents in any reasonable ranking scheme. Documents outside this obviously relevant

class may not be so reliably ranked. If this supposition is correct, doubling the size of the collection, will probably double the size of the obviously relevant class, leading to higher early precision. At present, this is mere conjecture and needs to be tested.

Data structure building time for the 4 gigabyte case is a little over double the comparable figure for the 2 gigabyte collection. Given the PADRE super dictionary architecture, it would be expected that index and dictionary building time would rise linearly with the amount of data but that building the super dictionary would require approximately three times as much I/O. Since super dictionary building for this size of collection takes only a fraction of the index building time, the observed ratio is quite within the range of expectation.

Query processing time increased by 53% when moving to the larger collection. An 80% increase might have expected on the basis of the increase from 5 to 9 in the number of superdictionary components. However, the fact that the CPU load of processing the additional four SD components is spread over 128 nodes rather than just 98 is a compensating factor. There are also some fixed costs, such as resetting at the beginning of a topic and returning rankings at the end, which are independent of the amount of data.

Disk space requirements are rather large because the HiDIOS filesystem [11] is organised for time efficiency rather than for minimising space. Significant imbalance between nodes in the DB Merging layout and the fact that 30 nodes have no data at all in this layout mean that parallel files include large amounts of unused space. In fact, the raw text for CD2 and CD4 occupies a total of 3.56 gigabytes in this layout! The fact that space requirements do not rise in proportion to the amount of data is indicative of the fact that the CD1/CD3 data is much better balanced on the machine.

8 Overall Summary and Conclusions

Taking into account the increased difficulty of the TREC-5 task and the fact that no query augmentation was employed, ANU/ACSys techniques for automatic query generation performed well and should constitute a sound base for eventual top-rank performance when coupled with a good query expansion system.

The span scoring model is capable of top-flight precision-recall results, subject to the use of good queries. Methods for refining span queries using term-term implications and pseudo relevance feedback conferred benefit but not as much as had been hoped. Taking into account the difficulty of the task and the use of interactive query development by most other groups, the ANU/ACSys manual adhoc queries performed quite well despite reduced development times and avoidance of subject-expert consultation. Clearly, the non-interactive manual approach to query formulation is unrealistic and an interactive framework for span-query development is an obvious direction for future PADRE development.

Further work is needed to confirm the validity of the techniques used to overcome OCR degradation. ANU/ACSys results in this category were hampered by lack of time to develop good queries, to systematically observe characteristic errors and to work with the heavily degraded dataset.

The DB Merging track provided a springboard for an extensive series of experiments which will be reported elsewhere. The use of the Fujitsu AP1000 to simulate the operation of distributed information servers is an interesting application of large-scale parallelism in IR.

The benefits of parallelism would be expected to show themselves most strongly in the VLC track. Good scalability was demonstrated by the ANU/ACSys VLC submissions. However, the two-fold increase in data size in the TREC-5 pre-track was too small to fully test the potential benefits. Nonetheless, experience gained in the pre-track has suggested ways to ensure better performance and scalability over the planned 20 gigabyte corpus. The observed increase in average precision as data size increased needs confirmation and, if necessary, explanation.

Acknowledgements

The cooperation of our colleagues at the University of Waterloo in making available their TREC-4 queries is gratefully acknowledged.

References

- [1] Peter Bailey and David Hawking. A parallel architecture for query processing over a terabyte of text. Technical Report TR-CS-96-04, Department of Computer Science, Australian National University, <http://cs.anu.edu.au/techreports/1996/>, 1996.
- [2] Chris Buckley, Amit Singhal, Mandar Mitra, and Gerard Salton. New retrieval approaches using SMART: TREC 4. In Harman [4], pages 25–48.
- [3] C. L. A. Clarke, G. V. Cormack, and F. J. Burkowski. Shortest substring ranking multitext experiments for TREC-4. In Harman [4].
- [4] D. K. Harman, editor. *Proc. Fourth Text Retrieval Conference (TREC-4)*, Gaithersburg, MD, November 1995. U.S. National Institute of Standards and Technology.
- [5] David Hawking. Document retrieval in OCR-scanned text. In *Proc. Sixth Parallel Computing Workshop*, Kawasaki, Japan, November 1996. paper P2-F.
- [6] David Hawking and Peter Bailey. *PADRE v. 2.4 User Manual*. http://cap.anu.edu.au/cap/projects/text_retrieval/.
- [7] David Hawking and Paul Thistlewaite. Methods for information server selection and result merging. In preparation.
- [8] David Hawking and Paul Thistlewaite. Proximity operators - so near and yet so far. In Harman [4], pages 131–143.
- [9] David Hawking and Paul Thistlewaite. Relevance weighting using distance between term occurrences. Technical Report TR-CS-96-08, Department of Computer Science, Australian National University, <http://cs.anu.edu.au/techreports/1996/index.html>, 1996.
- [10] S. E. Robertson. On term selection for query expansion. *Journal of Documentation*, 46(4):359–364, 1990.
- [11] Andrew Tridgell and David Walsh. The HiDIOS filesystem. In *Proc. Fourth International Parallel Computing Workshop*, pages 53–63, London, September 1995. Imperial College/Fujitsu.

Appendix A: Topic-by-Topic Performance of Best Distance-Based Queries

topic	ret	rel.ret/rel	%best	%median	@10	ave_prec	%best	%median
202	211	158/283	62%	87%	10	0.4952	68%	162%
203	102	24/33	83%	126%	4	0.2707	82%	347%
204	466	113/397	38%	55%	4	0.1215	34%	93%
205	1000	198/310	99%	1320%	10	0.3929	79%	14552%
206	1000	25/47	83%	312%	2	0.0597	46%	728%
207	1000	69/74	93%	100%	7	0.5805	90%	112%
208	1000	31/54	67%	111%	0	0.0514	21%	101%
209	1000	74/87	90%	101%	5	0.2711	77%	130%
210	1000	54/57	95%	108%	8	0.6084	79%	113%
211	1000	231/323	86%	186%	7	0.3781	66%	325%
212	1000	139/153	99%	145%	6	0.4514	80%	258%
213	57	16/21	80%	114%	6	0.4522	127%	203%
214	26	3/5	60%	60%	2	0.3462	51%	70%
215	1000	172/183	97%	108%	10	0.6154	98%	125%
216	512	34/36	97%	117%	7	0.5309	82%	129%
217	498	43/57	75%	154%	10	0.4620	76%	228%
218	12	7/46	17%	22%	5	0.0930	27%	89%
219	1000	96/133	99%	128%	3	0.1677	77%	172%
220	49	16/24	67%	73%	9	0.5226	80%	115%
221	1000	131/181	87%	116%	9	0.3112	64%	130%
222	74	44/74	64%	80%	10	0.5312	120%	211%
223	1000	147/363	57%	109%	6	0.1673	45%	163%
224	696	127/149	85%	134%	7	0.4229	88%	163%
225	1000	215/216	100%	119%	7	0.6985	96%	126%
226	1000	137/145	108%	326%	7	0.3888	94%	523%
227	1000	240/347	100%	189%	10	0.5097	107%	389%
228	1000	14/66	30%	56%	0	0.0066	6%	26%
229	424	20/21	95%	111%	7	0.5731	99%	154%
230	1000	82/85	98%	164%	10	0.6554	82%	495%
231	204	15/23	65%	68%	3	0.1631	70%	201%
232	186	5/9	71%	125%	1	0.1230	39%	1108%
233	280	110/121	101%	234%	9	0.6157	97%	1251%
234	57	26/28	96%	113%	9	0.7029	90%	166%
235	306	160/197	85%	150%	10	0.7171	89%	317%
236	1000	30/43	83%	333%	3	0.1080	104%	3484%
237	1000	180/215	97%	141%	8	0.4872	84%	203%
238	1000	220/270	86%	167%	5	0.3513	65%	316%
239	1000	69/123	77%	147%	3	0.1224	58%	334%
240	1000	173/276	88%	216%	8	0.2667	87%	516%
241	412	21/62	38%	131%	3	0.0798	25%	706%
242	396	33/38	89%	114%	9	0.5197	94%	374%
243	451	19/69	31%	50%	2	0.0350	15%	57%
244	1000	359/431	89%	120%	10	0.6201	94%	160%
245	1000	25/43	78%	147%	5	0.2021	93%	355%
246	1000	177/286	98%	174%	9	0.3132	101%	208%
247	76	28/36	80%	122%	5	0.4073	76%	119%
248	1000	113/122	109%	365%	7	0.4241	76%	1140%
249	1000	32/53	67%	86%	6	0.2063	73%	140%
250	1000	73/86	91%	149%	4	0.2040	61%	273%
49	33495	4528/6501	83%	137%	6.3	0.3634	78%	189%

Appendix B: Topic Similarities Between TREC-5 and TREC-4

These similarities were used in run `anu5mrg1` in the Database Merging track.

TREC-5 topic	TREC-4 topics in related area
251	203, 218, 219, 242, 244, 246
252	205, 209, 212, 240
253	205, 226, 232, 239
254	210, 216, 224, 229, 239
255	228, 243, 249
256	205, 206, 215, 226, 238
257	210, 215, 224, 239, 250
258	212, 223, 235, 240, 248
259	206, 222, 250
260	205, 216, 217, 239, 242
261	202
262	213, 214, 229, 232
263	213, 216, 231, 243
264	227, 236, 240, 247
265	211, 221, 235, 242, 250
266	220, 228, 248
267	225, 238
268	246
269	212, 219, 244
270	210, 216, 231, 243
271	204, 208, 230, 237, 249
272	210, 224, 229, 239
273	213, 217, 225
274	230
275	210, 216, 231, 243
276	205, 206, 215, 226, 238
277	202, 227, 236, 240, 246
278	217, 224, 213
279	213, 217, 240, 249
280	228, 236, 240, 249
281	213, 216, 224
282	221, 236, 250
283	219, 244, 246
284	221, 222, 235, 240, 250
285	202, 246
286	203, 218, 242
287	222, 235, 250
288	214, 216, 224, 232, 239
289	210, 216, 224, 226, 241
290	211, 219, 230, 237
291	209, 226, 231, 247
292	209, 245
293	207, 227, 246
294	208, 220, 238, 245
295	206, 207, 227, 235
296	206, 207, 231, 232, 241
297	206, 207, 216, 235, 245
298	221, 250
299	219, 226, 227, 246
300	206, 207, 227, 235



Parallel Techniques For Efficient Searching Over Very Large Text Collections

B. Mamalis P. Spirakis B. Tampakas

Computer Technology Institute

Kolokotroni 3, 26221, Patras, Greece

Tel.: + 3061 220112, Fax.: + 3061 993973

E-mail: {mamalis, spirakis, tampakas}@cti.gr

Abstract

This paper mainly discusses the efficiency of PFIRE system, a parallel VSM-based text retrieval system running on the GCel3/512 Parsytec machine, as well as the effectiveness of the corresponding pre-existing serial FIRE system. Concerning PFIRE, the use of suitable data sharing and load balancing techniques in combination with specific pipelining techniques and with the capability of building binary and fat-tree virtual topologies over the 2-D mesh physical interconnection network of the parallel machine, leads to very fast interactive searching over the large scale TREC collections. Analytical and experimental evidence is presented to demonstrate the efficiency of our techniques. The corresponding conventional FIRE system was also used to measure the effectiveness (in terms of recall and precision) of several IR techniques (statistical phrase indexing, automatic statistical global thesaurus construction etc.) used over the TREC WSJ subcollection.

1. Introduction

Our work consists of two main different parts:

- (a) testing (in terms of recall and precision measures) of the previously existing (and appropriately modified for large scale data management) FIRE system over the TREC WSJ subcollection.
- (b) integration of the parallel extension of FIRE system (PFIRE) via new advanced parallel techniques (data sharing and balancing techniques, pipelining, fat-trees processing) and extraction of various efficiency results (speed-up, utilization, response times) over the WSJ subcollection and some multiples of it.

Due to the fact that the conventional FIRE system uses rather traditional IR techniques, without something novel to be pointed out (except from our automatic global thesaurus construction approach), we present it

(as well as the corresponding 'TREC results' analysis) quite briefly. On the contrary, we give emphasis (and we describe in detail) on the new techniques (as well as some old ones out of [1]) used for the efficient parallelization of FIRE over the GCel3/512 Parsytec machine. The application of these techniques (which can be strongly recommended for the parallelization of any text retrieval system using the Vector Space Model) over the TREC WSJ collection has led to very satisfactory efficiency results which are suitably analysed and presented (in tables and curves) at the end-section of the paper.

In the following two sections we give a brief description of the basic IR features of FIRE (section 2) as well as a brief presentation of the TREC experiments performed over the WSJ subcollection (section 3). The description of the PFIRE system and a short reference to the hardware platform used (the GCel3/512 Parsytec machine) are given in section 4. Finally in section 5 (which is the most important part of our work), we present the efficiency results of PFIRE over the WSJ collection, as well as the fat-tree processing experiments that were performed with use of the techniques described in [20].

2. The FIRE System

FIRE (Full Information Retrieval Engine, [2]), is a text retrieval system aiming at effective (in terms of recall and precision measures) management and retrieval over large scale document collections. It is based on the Vector Space Model (VSM, [4],[6],[14]) using several text features (terms, phrases, thesaurus classes) as document identifiers. In the following paragraphs we present the basic characteristics of FIRE system (giving emphasis to those used for TREC experiments).

2.1 The Indexing Process

The initial objective of FIRE's indexing processes is to extract the most significant words of each document,

and use them as index-terms with appropriate statistically evaluated weights for the corresponding documents.

Both the stopping and the stemming process are applied to all the words of every document. Stopping implies the rejection of the words that are very common in English language. This is done by using a specific stop-list ([7]) that contains the most common words of the language. Stemming implies the extraction of each word's stem, leading to the substitution of the whole set of words in the document collection by the, substantially smaller in size, set of word-stems in the collection. This is done by using the Porter's algorithm as presented in [7]. The resulting word-stems are statistically weighted, (separately for each document), by the inverse document frequency (IDF) method ([6]) which is based on both the frequency of each term in a specific document and the frequency of each term in the whole set of documents.

Finally, a categorization of all the terms according to their document frequencies (DF) is performed. As stated in [14] the most appropriate terms for indexing are those with document frequency $N/100 < DF < N/10$ where N is the number of documents in the collection. The terms with $DF > N/10$ (high DF terms) are appropriate for construction of phrases, and the terms with $DF < N/100$ (low DF terms) are the most appropriate for thesaurus classes' construction. In FIRE, the low DF terms are used both for term-indexing and thesaurus classes' construction and the high DF terms are used for the extraction of phrases. However, during TREC experiments we have used several varying corresponding threshold values (instead of '10' and '100') in order to gain the best possible behaviour of the whole statistical indexing scheme.

According to the above indexing process, each document is finally represented by a corresponding document-vector (following the Vector Space Model) containing three types (single-terms, phrases, thesaurus classes) of weighted identifiers. These document-vectors are then organized into corresponding inverted indices (separately for each type of identifier).

2.2 Phrase Extraction

The statistical phrase extraction method used in FIRE is similar to the one described in [5]. Following the corresponding general statistical phrase extraction model of the above reference one has to determine three basic parameters: the length of the phrase, the domain in which the searching is performed, and the maximum distance (proximity) of the phrase elements within the specified domain. In FIRE we have used the typical minimum values for the above parameters (Length = 2, Domain = sentence, Proximity = 1) leading to the extraction of adjacent pairs of words within a sentence.

Following the considerations of paragraph 2.1 we have mainly used the high-DF terms of the document collection as phrases' terms. Specifically (as it is defined in [5]), at least one term of the phrase must be a high-DF term (based on parameter DFh), whereas the document-frequency of the whole phrase itself must exceed a specific threshold (parameter DFp). The resulting phrases (and the corresponding weights) naturally vary according to the above threshold values.

2.3 Automatic Thesaurus Construction

The automatic construction of the thesaurus classes in FIRE, is performed using the following procedure which consists of three main steps:

(a) Documents' clustering

First, the similarities of all pairs of documents in the collection are computed forming the corresponding similarity matrix. This computation is performed by the application of the well-known cosine similarity function ([6]) on the two term-vectors of each document-pair.

Given the above similarity matrix and one similarity threshold, several clustering algorithms based on graph theory can be used in order to extract corresponding documents' clusters. In FIRE, an algorithm based on the iterative evaluation of Connected Components (CCs) of controlled size, is used. This algorithm is briefly described below:

Assume that:

D : the whole set of the N documents in the collection
 MINTH : the desirable min number of docs per cluster
 MAXTH : the desirable max number of docs per cluster
 STEP : the increment value for similarity threshold
 INTN : the initial similarity threshold
 STEPTH : the current similarity threshold

The algorithm:

```

INTN = 0 ;
STEPTH = INTN + STEP ;
while (D NOT emptyset)
  begin
    Compute the CCs of D using STEPTH as similarity threshold for each CC
  for each CC do
    begin
      Let n be the # of docs in the CC
      if (n < MINTH)
        Ignore CC and remove its documents from D
      else if (n > MAXTH)
        Ignore CC (set D remains unchanged)
      else
        Accept CC as a proper document cluster
    end
  end
end
  
```



```

    end
    Increase STEPTH by the value STEP
end

```

The above algorithm finally leads to document clusters of size (# of documents) between MINTH and MAXTH. Although the Connected Components algorithm may be considered much simpler compared to existing algorithms (e.g. see the complete-link algorithm in [8],[9]), it is quite efficient when it is coupled with the appropriate parameter values.

(b) Extraction of the thesaurus classes

When the document clusters have been determined, the thesaurus classes themselves can be generated (one class for each cluster). The simplest and most reliable approach is to define a thesaurus class as the intersection of all the low frequency terms of the documents in a cluster. Alternate methods refer to the use of other set-operations such as union and restricted intersection. In FIRE an approach of extended intersection is used (extended, stands for the fact that this approach leads to classes with more terms than the simple intersection). It is supported by a user supplied parameter that determines the percentage of documents in a cluster that must contain one low frequency term in order for this term to be included in the corresponding thesaurus class.

(c) Expansion of the document term vectors by thesaurus classes

Once the thesaurus classes are generated, all documents in the collection can be indexed by "replacement" or "augmentation" ([8],[9]). Replacement entails replacing a term in a vector by a thesaurus class which contains that term. Augmentation involves adding the thesaurus class to the vector; the original term remains in the vector unchanged. We believe that augmentation is more effective than replacement. Another interesting point during the expansion of the term-vectors by thesaurus classes is the weighting of these thesaurus classes. Based on corresponding experiments, we conclude that the weighting function described in [8],[9] is quite effective (that weighting function has been used in FIRE with varying parameter values for down-weighting the resulted classes for each document).

2.4 The Retrieval Process

The documents' indexing process described in 2.1 is also used for indexing the queries. The query terms are filtered through the stop-list test and the stemming process, and then they are incorporated into the query term-vector (weighted by the standard value of

one). The query term-vector has the form of the document term-vectors. The expansion of the query term-vector by the appropriate phrases (extracted via the same method described in 2.2) and thesaurus classes (that contain at least one term of the query) directly follows.

The next step is to compare the query vector (following the Vector Space Model) to all the document vectors that represent the documents of the collection. The above comparison is based on the evaluation of the cosine-similarity value between the query and each document in the same way that the document to document similarity is evaluated in 2.3. The difference here is that the phrases and the thesaurus classes that have expanded both the query and the documents also take part in the query-document similarity computation. The final step is the ranking of the documents according to their scores (regarding the comparison to the query). The documents with the highest scores are then considered as the most relevant ones to the query.

2.5 Other Features

Two other significant features of FIRE system are the relevance feedback and passage retrieval methods used. The Ide dec-hi formula (see [12]) has been used for interactive relevance re-formulation of the queries, whereas the passage indexing and retrieval method used in FIRE is similar to those described in [10],[11]. The above characteristics of FIRE will not be described in more details since they have not been used for TREC experiments.

Also, FIRE is supported by a well-designed, windows-based (X-Windows, OSF/Motif), user interface. The user interface is generally seen as a key component to successful development, affecting important factors like the long-term durability of a software product. We studied the user interface of several known text retrieval systems like the SMART ([15]), the BASIS and the ARCHEION systems (the last two systems are commercial information retrieval products) and we have incorporated their most important key features in our implementation.

Another feature is the capability of defining categories over the whole document collection. Consequently, users are capable of selecting specific categories that will then participate in the retrieval process. Thus, the users are capable of directing the whole retrieval process closer to their needs. Finally, suitable distributed IR techniques have recently applied on FIRE system, providing the capability of efficient operation over highly distributed environments (see [3]).

3. TREC Experiments over FIRE

The retrieval effectiveness (in terms of recall and precision measures) TREC experiments that have been per-

formed over FIRE system aimed at mainly testing the basic IR features of FIRE over large document collections. In the following we briefly present these experiments and a draft analysis of the corresponding results.

3.1 Description of Experiments

The experiments have been performed over the:

- Ad-hoc topics (251-300, Description fields only)
- Category B data that consist of the WSJ subcollection (about 75000 documents, 250 Mbytes)

We've performed two official Ad-hoc automatic experiments (Ctifr1 and Ctifr2 runs) in which the following features of FIRE system were used for both the documents and the queries (the relevance feedback and passage retrieval methods have not been used yet for TREC data):

- stopping, stemming
- statistical word indexing
- statistical phrase extraction
- automatic global thesaurus construction

In both the experiments we tried to use suitable combinations of all the parameter values that had to be specified (high-DF and low-DF thresholds for phrase and thesaurus construction, special parameters for thesaurus construction - MINTH, MAXTH - etc.) The following parameter values were used for each run:

<u>Ctifr1</u>	<u>Ctifr2</u>
High-DF threshold: $\frac{N}{10}$	High-DF threshold: $\frac{N}{4}$
Low-DF threshold: $\frac{N}{1000}$	Low-DF threshold: $\frac{N}{500}$
MINTH: 20	MINTH: 10
MAXTH: 50	MAXTH: 30

(Ctifr2 parameter values lead to fewer phrases and smaller and more tight thesaurus classes than Ctifr1 run)

3.2 Statistics of the experiments

The serial experiments were performed on a (shared) SUN/Sparc Station 10 with 110 MHz, 32 MBytes main memory and 2 GBytes disk storage.

The total size of the inverted indices built for all the types of identifiers (terms, phrases, thesaurus) was 220 MBytes for Ctifr1 and 180 MBytes for Ctifr2.

Time spent for the preprocessing phase:

- 5.5 hours for Ctifr1 (2.5 hours for terms and phrases, 3 hours for thesaurus construction)
- 4.5 hours for Ctifr2 (2 hours for terms and phrases, 2.5 hours for thesaurus construction)

Average searching time per query : 60 seconds.

3.3 Analysis of Results

Considering the official testing of the conventional FIRE system over the WSJ collection the following remarks should be pointed out:

- Generally, the official recall/precision based results for both runs were not very good (specially concerning the average precision values over all the description topics), although that FIRE system has been tested successfully (with quite good recall/precision measures) over the old, very small in size, standard collections (Cranfield and Time).
- The results for Ctifr2 (fewer phrases and small and tight thesaurus classes seem to be a little better than the Ctifr1 results).
- There were some "very bad" topics that affected significantly the recall capability of the system (namely the ad-hoc topics 251, 264, 269, 270, 273, 274, 285, 291, 294 which need either tokenizer and syntactic analysis or thesaurus-of-synonyms support).

Specifically, in a total of 45 topics that had at least one relevant document we had (in Ctifr1 run): 1064 relevant docs in the collection - 378 relevant docs retrieved that is divided to:

- 1) 474 rel docs totally - 296 rel docs retrieved, for the 36 good queries and
- 2) 590 rel docs totally - 82 rel docs retrieved, for the 9 "very bad" queries

- However, the above bad queries did not affected so significantly the average precision values.

Trying to explain the relative retrieval failure of our experiments we could say that (beyond the large size of the collection and the advanced philosophy of the corresponding topics).

- (a) the relevance feedback and passage retrieval methods of FIRE system were not used (due to the lack of time)
- (b) there are several significant features (especially for TREC runs) that our system misses such as:
 - i. Syntactic documents' parsing and syntactic phrase indexing
 - ii. Tokenizer for dates, phone numbers etc. in documents
 - iii. Externally built auxiliary files (thesaurus)
 - iv. Advanced automatic query extraction, formulation and re-formulation methods (that seems to be the most important for TREC experiments)

4. The Parallel Extension of FIRE System (PFIRE)

The parallel extension of FIRE system has been based on a quite efficient VSM-based parallel binary-tree algorithm ([1]) which has been appropriately implemented on the GCel3/512 massively parallel machine. Generally, a lot of parallel text retrieval attempts have been done in the past with use of various hardware platforms and text processing models (e.g. [17],[18],[19]). The development and implementation of PFIRE resulted to a quite friendly and very efficient multiprocessor text retrieval system, preserving all the IR features of the conventional FIRE system as well as the X-Windows graphical user interface of FIRE. In the following paragraphs we briefly present the basic components of PFIRE (the hardware platform and the parallel algorithm).

4.1 The GCel3/512 Parsytec Machine

The hardware that we use in our implementation is the Parsytec GCel3/512 machine ([13]) which belongs to the MIMD class of parallel computers. It is a massively parallel machine consisted of 512 processor-units formed in a two dimensional grid interconnection network with dimensions 32 x 16. Each processor-unit is an Inmos T805 transputer, a member of the well known transputer family, which has the following characteristics:

- 32 bit RISC processor with frequency 30 MHz and peak performance 4.3 MFLOPS, 30 MIPS
- 4 KBytes on-chip cache memory and 4 MBytes external DRAM local memory
- 4 bidirectional high speed links of max data rate: 20 Mbits/sec (unidirectional) and 18.8 Mbits/sec (bidirectional)

The 512 transputer processors of the GCel3/512 computer are organized in a hierarchical structure. Each transputer is a processing unit. Eight processing units form a processing board. Two processing boards compose a cluster. A cluster (containing 16 transputers) is the atom of the GCel3/512 machine and it's the smallest unit that can be accessed by a user. Four clusters make a GigaCube and eight gigacubes make the whole GCel3/512 machine. From the user's point of view the whole 512-transputer network is split up in partitions of varying sizes. Each partition consists of one or more clusters. A partition may contain parts of other partitions.

The two-dimensional grid network of transputers is connected to the outside world through 2 external hosts which provide 12 high speed links (20 Mbits/sec). The hosts are two Sun/SPARC stations running the PARIX operating system. PARIX (PARAllel extensions to UNIX) is a UNIX-based operating system with extensions (tools and programs) that support the transputers'

communications with each other. PARIX is responsible for the efficient usage of the transputers' grid network.

4.2 The VSM-Based Parallel Algorithm

The main objective of the VSM-based parallel algorithm ([1]) used in PFIRE is to appropriately minimize the total communication times needed for the whole execution. Towards this direction, we've developed a virtual binary-tree processors' organization scheme over the 2-D mesh structured interconnection network of the machine. A brief description of the algorithm directly follows:

We suppose that we are given a complete binary-tree network of P processors where all of them ($(P+1)/2$ leaves and $(P-1)/2$ internal nodes) serve as working processors. Also we are given a VSM-indexed text query (query vector q_i) and a VSM-indexed text collection of D documents (and D document vectors as a consequence) which is appropriately shared over the local memories of the P processors (let's say approximately D/P vectors for each processor). Also there is a host processor which connects the root of the tree to the outside world and initially holds the query vector q_i . Under the above settings, the binary-tree single-query algorithm briefly consists of the following steps:

1. The query vector q_i is sent to all the processors of the tree progressively. Beginning from the root, each processor sends it to its two children and so on, till the query vector arrives at the leaves after $\log P$ steps (plus 1 step for the host-to-root communication).
2. Each processor performs the scoring and ranking subtasks over its own approximately D/P document vectors, thus resulting to one ranked relevant documents' set (RD) of size R for each processor.
3. The local ranked RD sets are merged progressively through all the levels of the tree (in parallel for the processors of the same level). Beginning from the lowest level (the parents of the leaves) each processor merges the RD sets of its 2 children into his own RD set, sends the result (of size R) to its parent and so on, till one final RD set is extracted by the root processor after $\log P$ such merging steps. This final set is then sent to the host processor (1 more step).

As it comes out of [1],[20], the total time spent for the execution of the above algorithm can be given by the following formula:

$$T = (2\log P + 2) + (T_s + T_{lr}) + 4R(\log P - 1) \quad (1)$$

where (a) $(2\log P + 2)$ is the total number of communication messages that are exchanged during the execution of all the phases of the algorithm ($\log P + 1$ from step

1 and $\log P + 1$ from step 3), (b) $(T_s + T_{lr})$ is the total computational time needed and it is almost equal to $\frac{1}{P}(t_s + t_{lr}) - t_s, t_{lr}$ represent the serial times for scoring and ranking respectively, and (c) $4R(\log P - 1)$ is the merging time ($4R$ for merging of 3 RD sets in one node and $\log P - 1$ because this task takes place sequentially in all the $\log P - 1$ levels of the tree).

Concerning the performance of the above algorithm, we can observe that the communication times remain logarithmic, whereas the computational tasks are parallelized almost perfectly (see [1] for further explanation). The corresponding experimental results (presented in [1]) gained by the application of the algorithm over the GCel machine (via the use of virtual tree topologies over the physical 2D-mesh network of the machine) validate a very efficient behaviour over the, small in size, CRANFIELD and TIME text collections and over simulated large scale text collections (formed by reproducing the CRANFIELD collection multiple times). In the next section we demonstrate the outstanding efficiency of the above algorithm (extended by data balancing and pipelining methods) even when used over the large scale TREC WSJ subcollection.

5. TREC Experiments over PFIRE

5.1 Description of Experiments

Various efficiency TREC experiments have been performed over PFIRE towards the direction of gaining very fast interactive response times over a large scale "real" text collection. The previously (4.2) mentioned binary-tree algorithm (extended by new parallel techniques described in 5.2) was used, as well as our new multiple-query fat-tree algorithm ([20]). All the experiments have been performed over the GCel3/512 Parsytec Supercomputer (4.1). The collection data used were

- The TREC WSJ subcollection (about 250 MBytes)
- Parts (1/2 and 1/4 WSJ) and simulated multiples (up to 8-times WSJ) of the WSJ subcollection (by reproducing the WSJ appropriately) which result up to 2 GBytes of indexed-data

whereas the Ad-hoc TREC topics (251-300) have been used as user queries in order to obtain the desirable average performance measurements per query for all the experiments.

5.2 Modifications on the Binary-tree Parallel Algorithm

As mentioned above (and clearly pointed out in [1]) the existing binary-tree parallel algorithm gives very good results (up to 80% utilization and very fast response times) in the general case and was expected to give (via

simulations) excellent results when the data are uniformly distributed over the processors of the machine (up to 90% utilization). Trying to gain even better results (for both response times and speed-up and utilization measures) over "real" large collection data we've performed the following challenging modifications:

(a) Data sharing and balancing

Obviously (with regard to the binary-tree algorithm of [1]), the way that the D document vectors of the whole collection are shared to the P processors of the machine is quite critical because it directly affects the distribution of the computational load (which consists of the local scoring and local ranking tasks) over the P processors. Moreover, it's obvious that if the computational load of even one processor is much larger than the computational load of the other processors, it could result to significant undesirable increase of the total computational times $(T_s + T_{lr})$ due to the fact that the merging phase of the algorithm requires the completion of the local scoring and ranking tasks of all processors, in order to be correctly terminated.

Generally, considering the above problem (and having in mind that the computational load of each processor consists of the local scoring and ranking tasks) we can state the following basic requirements concerning the desirable data-sharing and balancing scheme of the D document vectors over the P processors of the machine:

- From the "local ranking" point of view, the most preferable solution requires each processor to process the same number of document vectors (let's say approximately $\frac{D}{P}$ vectors).
- From the "local scoring" point of view, the most preferable solution requires each processor to process the same amount of data in it's own local memory (the sum of the sizes of it's own vectors), independently to the number of these vectors).

Obviously, it's quite easy to satisfy the first requirement by randomly loading $\frac{D}{P}$ vectors to each processor (e.g. the first $\frac{D}{P}$ vectors to the first processor etc.). However, the satisfaction of the second requirement implies the use of a time-consuming sorting-based algorithm. Moreover, it is very hard for any existing algorithm to satisfy optimally both the above requirements considering the usual case of a text collection with varying, in size, document vectors (there is an obvious contradiction between the two requirements - the achievement of equal amount of data on each processor normally leads to different number of vectors for each processor).

In the initial implementation of PFIRE we had followed the easy solution of satisfying the first requirement only, having the advantage of the direct (not time-consuming) loading of the processors of the ma-

chine, and the disadvantage of undesirable computational overheads due to the varying sizes of the document vectors of the VSM-indexed text collection. On the contrary, during TREC experiments we tried to develop a data-sharing and balancing scheme that satisfies both the above requirements (as much as possible) towards the direction of gaining faster response times and even better speed-up and utilization measures.

Our data-sharing scheme is naturally based on the initial sorting of all the document vectors according to their sizes. Afterwards, it tries to distribute the sorted document vectors over the P processors in such a way that each processor finally process the same number of vectors and almost equal amount of data, providing that the variations on the sizes of the different vectors are of normal extent. The key idea of this specific distribution is to serially divide the sorted set of vectors on subsets of size P , and then share each subset (by one-to-one mapping) to the P processors in such a way that each time the corresponding subset is mapped to a different partition of the processors' set.

More specifically, given a VSM-indexed (with any number of different types of identifiers – words, phrases and thesaurus classes in our case) text collection of D documents (that leads to D corresponding document vectors) and P processors as an ordered set $P_{set} = (1, \dots, P)$, our off-line data balancing algorithm can be described (in a more formal way) as follows:

- A. Sort the D document vectors according to their size, thus leading to an ordered set $D_{sort} = (D_1, \dots, D_i, \dots, D_D)$, where D_1 is the vector of the maximum size and D_D denotes the vector of the minimum size.
- B. for $i = 0$ to $(\lfloor \frac{D}{P} \rfloor - 1)$
 - 1: Distribute the $(D_{1+(i \times P)}, \dots, D_{(i+1) \times P})$ vectors of D_{sort} to the processors of P_{set} by one-to-one mapping (e.g. $(D_{1+(i \times P)})$ is given to the first processor of P_{set} etc.
 - 2: Shift P_{set} one position right (e.g. with one shift P_{set} becomes $(2, \dots, P, 1)$)
- C. Distribute the remaining $(D_{1+(\lfloor D/P \rfloor \times P)}, \dots, D_D)$ vectors of D_{sort} to the first $D - (\lfloor D/P \rfloor \times P)$ processors of the last shifted P_{set}

Obviously, the above algorithm leads to equal number of vectors for each processor. Beyond this, with regard to the requirement of giving almost the same amount of data to each processor, it's clear that the effectiveness of our scheme is collection dependent. However, it clearly promises much better results (concerning the response times, speed-up and utilization measures) comparing to the method that was initially used in PFIRE. Now, considering the TREC collections, we can say that the cor-

responding variations on the sizes of the document vectors are quite extended, however they are not extreme. Moreover (as it is clearly validated by the experimental results presented in section 5.3), the improvements offered by the use of our data-sharing method are quite significant and the undesirable computational overheads caused by the initially used random data distribution are properly eliminated.

(b) Pipelining

The pipelining techniques that have been incorporated into the binary-tree algorithm aim at the appropriate minimizing of the average query waiting times when multiple queries are to be processed together (the one after the other) by the P processors of the machine. Specifically, supposing that we are given X user queries that are to be treated together by our binary-tree algorithm (the one after the other) we introduce the following pipelined steps:

1. *During the construction and the propagation of the user query through the tree.* Specifically, the host processor is allowed to construct and send sequentially all the queries that have arrived to him, to all the processors independently to the completion of the remaining tasks for each query. It's obvious that the total time needed for the execution of the above pipelined task for the propagation of all the X queries is equal to $\log P + X$ message steps ($\log P/X + 1$ in average for each query), whereas without pipelining we had to spend $\log P + 1$ message steps for the propagation of one query only.
2. *During the merging phase.* Specifically, each processor, after the completion of the local scoring and ranking tasks for one query, is allowed to continue by performing local scoring and ranking for the next query, independently to the completion of the merging task of the first query. Obviously, one processor can behave in such a way, either in the case that he waits for the RD sets of his two children or in the case that he has sent his RD set (merged with the RD sets of his two children) to his parent. Moreover, the above pipelining can be reliably be applied on the binary-tree algorithm since each processor receives all the X queries from the beginning (this is achieved via the previously mentioned pipelining on the construction and propagation of the user queries). We also have to note here that the implementation of such a pipelining technique requires very careful treatment because of the complicated synchronization that is introduced.

Now, considering the average waiting time improvements achieved per query with use of the above pipelining techniques (with regard to equation (1) and the cor-

responding discussion of section 4.2) we have to outline the following remarks:

- The logarithmic factor of the query propagation message steps is almost eliminated (we refer to the $\log P + 1$ query message steps out of the $(2\log P + 2)$ totally needed communication steps).
- The merging overhead (initially caused due to the need of the sequential execution of the merging phase through the $\log P$ levels of the tree) is reduced quite significantly. Actually, the logarithmic factor of the merging computational task (we refer to the factor $4R(\log P - 1)$ of equation (1)) is almost eliminated.

Moreover, considering the large size of the RD sets required in TREC experiments (equal to '1000', thus resulting to increased merging overhead), we can say that the corresponding improvements are expected to be quite clear. However, we have to note here that the logarithmic factor of the RD message steps (the half of the $(2\log P + 2)$ totally needed communication steps) can not be eliminated (which means that the logarithmic factor concerning the total communication times still remains). Actually, the complete elimination of the logarithmic factors of equation (1) (that implies the substantial minimization of the communication times of our algorithm) requires the use of more complex pipelined parallel algorithms running on higher capacity interconnection networks (than a simple binary-tree topology and the 2-D mesh structure of the GCel3/512 machine). For example, see [20] where a *fat-tree* is appropriately used or section 5.4 for a brief description.

The corresponding experimental results (presented in the following section) demonstrate the quite efficient behaviour of the above pipelining techniques and they validate all the above theoretically stated considerations.

5.3 Efficiency of Searching

In order to obtain a realistic evidence concerning the high performance of PFIRE system over the large scale TREC WSJ collection (both with and without the use of the data-sharing and pipelining techniques) we've performed various efficiency experiments (under the settings of section 5.1) for the following separate cases:

- with and without data sharing and balancing
- varying collection size
- varying number of retrieved documents (R)
- with and without pipelining

In all the above cases the corresponding measurements have been taken for varying number of processors (ranging from 16 to 512). The performance measures evaluated in each case are the well-known *speed-up* (S_P) and

utilization (U_P) measures which can be defined as

$$S_P = \frac{T_1}{T_P}, \quad U_P = \frac{S_P}{P}$$

where T_1 is the time to carry out the whole system's work on one processor and T_P corresponds to the time needed when P processors are used for the same work in parallel. In the ideal case, $S_P = P$, whereas the utilization measure denotes (practically) the fraction of the ideal speed-up that is achieved in each case. Also, the response times (R_P , in seconds) of the system are measured for all the experiments.

(i) Data sharing and Balancing

Concerning the worth of using the data sharing and balancing scheme described in 5.2, we present in tables 1 and 2 extended comparative experimental results gained by the execution of the binary-tree algorithm both with (table 2) and without (table 1) data-balancing. Specifically, in both the tables the response times and the speed-up and utilization measures are given for varying number of processors (from 16 to 512) and for varying collection size. With regard to the collection size we have used the WSJ collection, a fraction of the WSJ of almost 60MB (1/4-WSJ) and a simulated multiple of the WSJ of almost 1.9GB (the 8-WSJ). Also, for both the experiments we assume that the value of R (# of retrieved documents) is equal to '1000'.

The first conclusion that can easily be extracted from tables¹ 1,2 is that the results gained by the use of data-balancing (table 2) are substantially better than the ones offered by the initial binary-tree algorithm of PFIRE. As an example, the utilization measure for 64 processors over the 1-WSJ collection increases to 90% (from 78% without balancing) and finally becomes 97% for 512 processors over the 8-WSJ (from 80%). Improvements of similar extent hold for the response times too (e.g. for 512 processors over the 1-WSJ collection we obtain response time 0.8 seconds with data-balancing and 0.99 seconds without balancing).

Moreover, the reader can easily notice that the corresponding improvements slightly increase (a) with the increase of the # of processors (e.g. the difference 90%-to-78% for the 1-WSJ and 64 processors becomes 80%-to-53% for 512 processors) and (b) with the increase of the collection size (e.g. the difference 38%-to-22% for 256 processors and the 1/4-WSJ becomes 85%-to-63% for the 1-WSJ). The former ((a)) can naturally be explained by the fact that for larger # of processors (and constant collection size which means constant # of vectors) the random sharing of the data-vectors initially used by PFIRE leads to worse distribution since each processor gets fewer document vectors.

¹The '-' in some cells means that the corresponding x-WSJ collection does not fit in the total memory of the P processors.

P	Speed-up			Utilization			Response Time		
	1/4 WSJ	1 WSJ	8 WSJ	1/4 WSJ	1 WSJ	8 WSJ	1/4 WSJ	1 WSJ	8 WSJ
16	11.2	—	—	70%	—	—	5.25	—	—
32	19.5	—	—	61%	—	—	3.12	—	—
64	32	49.9	—	50%	78%	—	1.78	2.51	—
128	46.1	89.6	—	36%	70%	—	1.11	1.79	—
256	56.3	161.2	—	22%	63%	—	0.89	1.32	—
512	54.8	271.3	408.8	14%	53%	80%	0.78	0.99	3.31

Table 1: Measurements for varying collection size without balancing

P	Speed-up			Utilization			Response Time		
	1/4 WSJ	1 WSJ	8 WSJ	1/4 WSJ	1 WSJ	8 WSJ	1/4 WSJ	1 WSJ	8 WSJ
16	12.9	—	—	81%	—	—	5.12	—	—
32	23	—	—	72%	—	—	2.87	—	—
64	39.6	57.6	—	62%	90%	—	1.61	2.39	—
128	61.4	112.6	—	48%	88%	—	0.97	1.62	—
256	97.2	217.6	—	38%	85%	—	0.68	1.13	—
512	158.7	409.6	496.6	31%	80%	97%	0.52	0.8	3.05

Table 2: Measurements for varying collection size with balancing

P	Speed-up			Utilization			Response Time		
	50	100	1000	50	100	1000	50	100	1000
64	59.2	58.7	57.6	92.5%	91.8%	90%	2.08	2.19	2.39
128	114.8	113.9	112.6	89.7%	89%	88%	1.37	1.44	1.62
256	224.2	222.7	217.6	87.6%	87%	85%	0.98	1.02	1.13
512	424.9	420.3	409.6	83%	82.1%	80%	0.7	0.73	0.8

Table 3: Measurements for varying # of retrieved documents with balancing

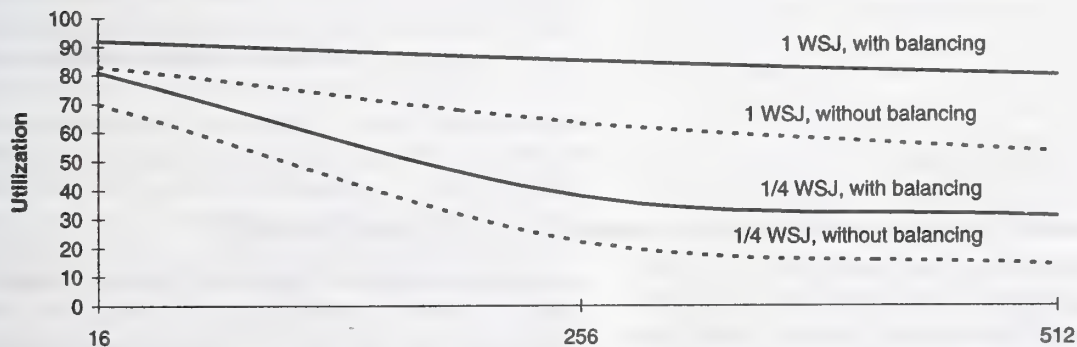


Figure 1: Utilization curve for varying # of processors with and without balancing

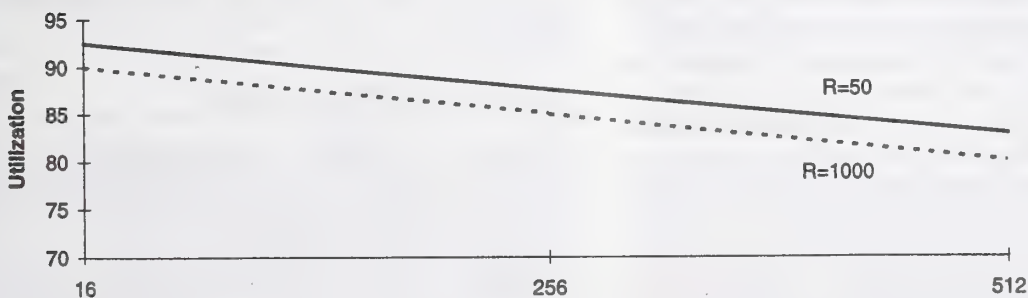


Figure 2: Utilization curve for varying # of processors and retrieved docs

P	Speed-up			Utilization			Response Time		
	1/4 WSJ	1 WSJ	8 WSJ	1/4 WSJ	1 WSJ	8 WSJ	1/4 WSJ	1 WSJ	8 WSJ
16	13.9	—	—	87%	—	—	4.83	—	—
32	25.2	—	—	79%	—	—	2.7	—	—
64	44.8	59.5	—	70%	93%	—	1.5	2.11	—
128	78	117.7	—	61%	92%	—	0.89	1.49	—
256	135.6	230.4	—	53%	90%	—	0.63	1.06	—
512	250.8	445.4	501.7	49%	87%	98%	0.49	0.75	3.01

Table 4: Measurements for varying collection size with pipelining

P	Speed-up		Utilization	
	with	without	with	without
16	12.9	11.2	81%	70%
32	27.2	24	85%	75%
64	57.6	49.9	90%	78%
128	119	101.1	93%	79%
256	245.7	203.5	96%	79.5%
512	501.7	408.8	97%	80%

Table 5: Measurements for full-of-data processors with and without balancing

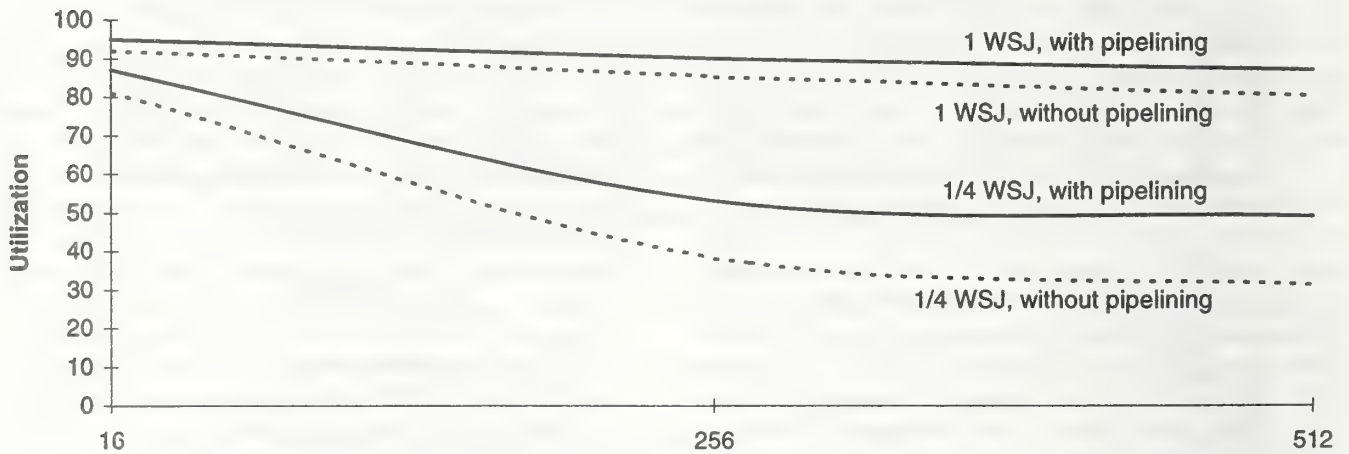


Figure 3: Utilization curve for varying # of processors with and without pipelining

Thus, the corresponding overhead grows up and the difference comparing to a suitably balanced indexed-data set tends to be substantial. On the other hand, concerning the *latter* ((b)) one would expect the opposite effect since the use of a larger collection implies more document vectors for each processor and the result of the random data-vectors sharing is expected to be quite better. However, when larger collections are used (as it is clearly explained in the directly following paragraphs) the positive effect due to the increase of the computational load for all the processors is much more critical and finally it dominates over any other factor. A quite clear view of the above conclusions is given also by the corresponding utilization curves presented in figure 1 (two pairs of curves, one for the 1/4 WSJ and one for the 1-WSJ collection). Finally, we have to note here (this note holds for all the experiments presented in

section 5) that although the achieved response times are very good, much better response times could be achieved if we had assumed lower values of R . Such experiments (for $R = 10$) which lead to excessively low response times for large # of processors can be found in [20].

(ii) Varying collection size

Based on the experimental results presented in both the tables 1 and 2 we can also extract some quite interesting conclusions concerning the influence of the varying collection size on the efficiency of our algorithm, independently to the use of the data sharing and balancing scheme. First, it's obvious that, for constant number of processors, the speed-up and utilization measures increase significantly with the increase of the collection size (e.g. for 128 processors and with the use of

data-balancing – table 2 – the utilization measure is 48% for the 1/4-WSJ collection and increases to 88% when the 1-WSJ collection is used. This obviously happens because the use of a large collection implies larger amount of data loaded to each processor which results to larger computational times. Consequently, the increased computational times dominate over the communication times (which remain constant for constant # of processors), thus leading to increase of the achieved speed-up and utilization values.

Following the same considerations (as they can be stated in a similar way for small, in size, collections) two other interesting observations coming out from tables 1,2 can be naturally explained: (a) all the speed-up and utilization values decrease significantly with the increase of the # of processors when the collection size remains constant and (b) we obtain quite low speed-up and utilization values when the collection size used is too small for the total memory capacity of the P processors (e.g. 31% and 14% utilization when the 512 processors and the 1/4-WSJ collection are used, with and without data-balancing respectively). As a general conclusion we can say that, given a constant number of processors, the best possible speed-up and utilization values would be achieved in the case that all the processors are full of indexed data. Again, the utilization curves presented in figure 1 can serve as a more clear demonstration of all the above conclusions.

Furthermore, in order to obtain additional evidence concerning the grate positive effect of the large collection size over the performance of our algorithm (as well as for the outstanding efficiency of the algorithm itself in that case) we have performed another very interesting experiment (see table 5) which outlines the speed-up and utilization measures for various values of P (with and without data-balancing), providing that in each case the P processors are full of data. The corresponding parts and multiples of the WSJ collection that have been used in order to fill the P processors with indexed-data are the following: 1/4-WSJ for 16 processors, 1/2-WSJ for 32, 1-WSJ for 64, 2-WSJ for 128, 4-WSJ for 256 and 8-WSJ for 512 processors).

As it comes out from table 5, the values for both the speed-up and utilization measures are very high for all the varying values of P . As an example, when data-balancing is used the utilization values start at 81% for 16 processors and raise up to 97% for 512 processors. Also, we can observe that in this case (full of data processors) the utilization values increase with the increase of P (as it is opposed to the case of using the same collection size for all the different numbers of processors – see tables 1,2). The above observation proves that the collection size is the most important factor concerning the performance of our parallel algorithm. Additionally (as it was expected), both the speed-up and utilization

are quite better when data-balancing is used. However, the values for the case that data-balancing is not used are also very satisfactory. Finally, in the figure next to table 5, the reader can find the corresponding utilization curves which lead to the same conclusions referred above. Both the curves start at very high point-values and slightly raises up to even higher values (as opposed to the corresponding curves of figure 1).

(iii) Varying number of retrieved documents

As mentioned above, all the measurements of tables 1 and 2 have been taken for value of R (# of retrieved documents) equal to '1000' (as it is required for TREC experiments). Obviously, this large value of R causes additional computational overheads because it directly affects the total merging time needed for the execution of the algorithm (see equation (1)). Thus (in order to demonstrate the corresponding difference) we present in table 3 some experimental measurements for more "realistic" values of R ('50' and '100') and we compare them to the ones (included in table 3 too) for $R=1000$. All the measurements of table 3 have been performed for the 1-WSJ collection and with use of data-balancing.

As it comes out directly from table 3 all the measured values (speed-up, utilization, response times) become quite better for $R=50$ or $R=100$ (comparing to the values for $R=1000$). Moreover, we can observe that the utilization measure for $R=50$ and 64 processors (where the 1-WSJ collection fits exactly) raises up to 92,5% (quite significant improvement comparing to the 90% utilization value for $R=1000$). The corresponding differences can also be observed out of the utilization curves for $R=50$ and $R=1000$ that are presented in figure 2. We have to note here, that all the differences (specially concerning the speed-up and utilization measures) would be even more significant if the data-balancing technique was not used. As a general conclusion (considering the results of table 3 and figure 2) we could say that for smaller (and more realistic) numbers of retrieved documents our algorithm (including data-balancing) becomes even more efficient and it leads to very high utilization values even when the processors of the machine are not full of data (e.g. almost 90% for $R=50$, 128 processors, 1-WSJ collection).

(iv) Pipelining

Finally, in table 4, we present the experimental results gained by allowing pipelining during the merging and the query propagation phases of our algorithm. Generally, table 4 has the same form with the previously examined tables 1,2, whereas all the measurements presented in this table have been taken for value of R equal to '1000' and with use of data-balancing (thus,

P	Speed-up			Utilization			Response Time		
	1/8 WSJ	1/4 WSJ	1 WSJ	1/8 WSJ	1/4 WSJ	1/2 WSJ	1/8 WSJ	1/4 WSJ	1/2 WSJ
8	6.64	—	—	83%	—	—	4.46	—	—
16	12.16	13.92	—	76%	87%	—	2.65	4.83	—
32	22.08	25.28	25.92	69%	79%	81%	1.6	2.7	4.03
64	38.4	44.8	46.72	60%	70%	73%	0.89	1.5	2.71

Table 6: Measurements for the binary-tree algorithm

P	Speed-up			Utilization			Response Time		
	1/8 WSJ	1/4 WSJ	1/2 WSJ	1/8 WSJ	1/4 WSJ	1/2 WSJ	1/8 WSJ	1/4 WSJ	1/2 WSJ
8	6.72	—	—	84%	—	—	4.23	—	—
16	12.64	14.24	—	79%	89%	—	2.35	4.61	—
32	23.68	26.56	27.2	74%	83%	85%	1.39	2.47	3.68
64	42.88	48	50.56	67%	75%	79%	0.73	1.31	2.39

Table 7: Measurements for the fat-tree algorithm

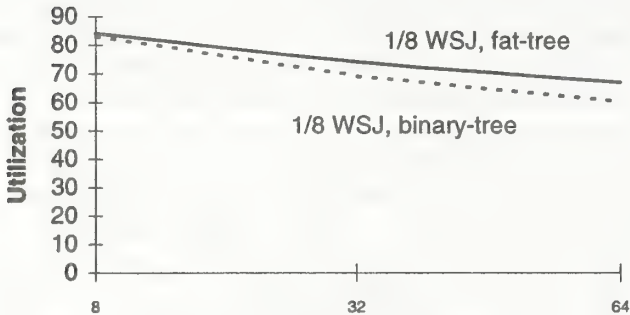


Figure 4: Utilization curves

relative comparisons should be done with table 2).

More specifically, by comparing tables 2 and 4, we observe an obvious improvement (concerning all the performance measures – speed-up, utilization, response times) offered by the use of our pipelining techniques (table 4). Moreover, the reader can notice that this improvement slightly increase with the increase of the number of processors (P) of the machine. This happens because the corresponding time-saving (the logarithmic factors, as it is explained in 5.2) is affected almost directly by the logarithm of P . The above conclusion can be extracted more clearly by observing the corresponding utilization curves in figure 3 (two pairs of curves, for the 1-WSJ and the 1/4-WSJ collection respectively). Additionally, as it comes out from figure 3 the improvements offered by the pipelining are larger when the smaller collection is used (the 1/4-WSJ collection). This happens because for smaller collections (that means less computational time for each processor) the corresponding time-saving due to pipelining on the query propagation phase (which is a part of the communication times) is more critical.

Another significant conclusion coming out from table 4 is that the combination of our data-balancing and pipelining techniques lead to the best possible experimental results for our system in all cases (under the restriction of $R=1000$).

The utilization measure raises up to 98% for 512 processors and the 8-WSJ collection, whereas the response time becomes 0.75 seconds for 512 processors and the 1-WSJ collection.

5.4 Multiple Queries Processing via Fat-trees

Beyond the extended experiments performed for the binary-tree algorithm of PFIRE we've also tried to obtain some realistic experimental results concerning our new fat-tree multiple-queries parallel algorithm ([20]). Briefly, the above algorithm, assuming an ideal fat-tree underlying interconnection network of P processors and a set of X concurrently arriving queries, it processes all the queries together and achieves excessively low total communication times (almost constant) by taking advantage of the high-capacity channels of the ideal fat-tree topology. Moreover (see [20]), the corresponding total processing times (both the amortized and the average waiting times) are proved, both analytically and experimentally over the TREC WSJ collection, to be substantially better than the ones offered by the binary-tree algorithm, provided that the two algorithms run on almost ideal fat-tree and binary-tree underlying networks. We achieve this in [20] by developing equivalently efficient embeddings (concerning the fat-tree we use a reduced version of the corresponding embedding in order to have a fair comparison) of the two topologies over the 2-D mesh network of the GCel Parsytec machine (see [20]). The complete presentation of the algorithm as well as the detailed analysis of its' performance can be found in [20]².

Via the following experiments (tables 6,7) we examine the case of using the full-version of our embedding method ([20]) (which is quite unfair for the fat-tree algorithm due to the fact that its' high-capacity channels

²This reference can easily be found by contacting B. Mamalis to mamalis@cti.gr.

is impossible to be efficiently simulated by a 2-D mesh). In this case and for large values of P (e.g. greater than 128) the fat-tree algorithm can not offer better results than the binary-tree algorithm. However, for # of processors up to 64 all the corresponding experimental measures are clearly better. These experimental results are presented in table 6, whereas in table 7 the corresponding results for the binary-tree algorithm are given (with data balancing and pipelining) in order the relative comparisons to be easily performed³. Both the above tables are of the same type with tables 1,2 (and the value of R is equal to '1000') with the difference that here, we have used smaller in size collections (1/8, 1/4 and 1/2 WSJ) because of the smaller values of P .

Beyond the relative (not substantial, however clear) improvements offered by the fat-tree algorithm, the reader can easily notice that these improvements increase with the increase of P . This happens because as P increases the communication times of the fat-tree algorithm remain almost constant, whereas the communication times of the binary-tree algorithm increase logarithmically. A more clear view of the above observation is given in figure 4 where the utilization curves for both the algorithms over the 1/8-WSJ collection are presented (the distance between the two curves increase with the increase of P). However, if we had corresponding experimental results for values of P greater than 64 we would observe that the fat-tree utilization measures would decrease progressively and the utilization curve would start to go down sharply (due to the extra overheads that are caused by the full-version of our embedding for large values of P).

6. References

- [1] P. Efraimidis, C. Glymidakis, B. Mamalis, P. Spirakis, B. Tampakas, "Parallel Text Retrieval on a High Performance Supercomputer Using the Vector Space Model", ACM SIGIR'95 Conference, July 9-13, Seattle, pp. 58-66, 1995.
- [2] M. Lafazanis, B. Mamalis, P. Spirakis, B. Tampakas, and A. Tsakalidis, "FIRE: A Flexible Tool for Efficient Information Retrieval", RIAO '94 Conference, New York, October 11-13, 1994 (accepted for prototype demonstrations).
- [3] K. Antonis, B. Mamalis, B. Papakostas, P. Spirakis, A. Stamoulis, B. Tampakas, "Effective Distributed Information Retrieval Techniques with the Vector Space Model", ISCA PDCS'96 Conference, September 25-27, Dijon, France, pp. 726-731, 1996.
- [4] G. Salton, "Automatic Text Indexing Using Complex Identifiers", ACM SIGIR'88 Conference, pp.135-143, 1988.
- [5] J.L. Fagan, "Automatic Phrase Indexing for Document Retrieval: An Examination of Syntactic and Non-Syntactic Methods", ACM SIGIR'87 Conference, pp. 91-99, 1987.
- [6] G. Salton and McGill, "An Introduction to Information Retrieval", 2nd edition, McGraw-Hill, c1983.
- [7] B. Frakes and R.B. Yates, "Information Retrieval: Data Structures and Algorithms", Addison-Wesley, c1992.
- [8] C.J. Crouch, "A Cluster-Based Approach to Thesaurus Construction", ACM SIGIR'88, pp. 309-320, 1988.
- [9] C.J. Crouch and B. Yang, "Experiments in Automatic Statistical Thesaurus Construction", ACM SIGIR'92 Conference, pp. 77-85, 1992.
- [10] G. Salton, J. Allan, C. Buckley, "Approaches to Passage Retrieval in Full Text Information Systems", ACM SIGIR'93, pp. 49-58, 1993.
- [11] M. Hearst, C. Plaunt, "Subtopic Structuring for Full Length Document Access", ACM SIGIR'93, pp. 59-68, 1993.
- [12] D. Harman, "Relevance Feedback Revisited", ACM SIGIR'92 Conference, pp. 1-10, 1992.
- [13] Frank Tiedt, "Parsytec GCel Supercomputer", Technical Report, Parsytec Computer GmbH, July 1992.
- [14] G. Salton, et al, "A Vector Space Model for Automatic Indexing", Communications of the ACM, 18 : 613-620, 1975.
- [15] Tong Li and V. Chiu, "X-Window Interface to SMART, an Advanced Text Retrieval System", 1992.
- [16] C. Leiserson, "Universal Networks for Hardware Efficient Supercomputing", IEEE Transactions on Computers, Vol. C-34, No. 10, October 1985.
- [17] J. Cringean, R. England, G. Manson, P. Willett, "Parallel text Searching in Serial Files Using a Processor Farm", ACM SIGIR'90, pp. 429-452, 1990.
- [18] F. Grandi, P. Tiberio, P. Zezula, "Frame Sliced Partitioned Parallel Signature Files", ACM SIGIR'92, pp. 286-297, 1992.
- [19] C. Stanfill, "Partitioned Posting Files: A Parallel Indexed File Structure for Information Retrieval", ACM SIGIR'90, pp. 413-428, June 1990.
- [20] B. Mamalis, P. Spirakis, B. Tampakas, "Optimal High Performance Parallel Text Retrieval via Fat Trees", CTI Technical Report, Submitted to SIGIR'97 Conference.

³Relative comparisons to the experimental results of [20] can not be done due to the use of different settings (e.g the value of R , different versions of the embedding etc.)



Document Retrieval Using The MPS Information Server (A Report on the TREC-5 Experiment)

François Schiettecatte
(francois@fsconsult.com)

FS Consulting, Inc.
1890 Highland Avenue, Rochester, NY, 14618
<http://www.fsconsult.com/>

1 Introduction

This paper summarizes the results of the experiments conducted by FS Consulting, Inc. as part of the Fifth Text Retrieval Experiment Conference (TREC-5). We participated in Category C, ran the ad-hoc experiments and participated in the database merging track, producing three sets of official results (fsc1t3, fsc1t4 and fsc1t3m) as well as some unofficial results (fsc1t4a). Our long-term research interest is in building information retrieval systems that help users find information to solve real-world problems. Our TREC-5 participation centered on two goals: to see if automatic query reformulation (relevance feedback) provides better results than the searcher's query reformulation; and to evaluate the effectiveness of the document scoring algorithms when searching across multiple databases. Our TREC-5 ad-hoc experiments were designed around a model of an experienced end user of information systems, one who might regularly use a system like the MPS Information Server while seeking information in a workplace or library setting

2 Overview of FS Consulting TREC-5 Experiments

In the TREC-5 experiments we set out to answer two questions:

- Does automatic query reformulation (relevance feedback) provides better results than the searcher's query reformulation?

We began with the assumption that our information seeker had previous experience using on-line retrieval systems commonly available in libraries (e.g., on-line library catalogs and bibliographic systems like MEDLINE). Although the search interface varies considerably, most library systems default to a novice-type search interface that allows a searcher to enter one or two terms and apply a Boolean operator to relate them. To aid the more

advanced end-user/searcher, academic libraries typically provide search aids and training sessions that teach searchers to construct more complicated Boolean statements and employ controlled vocabularies for term selection. Additionally, information seekers who patronize librarian-mediated search services have observational experience to draw upon: often, they are asked to create query statements, or to work directly with librarians performing the search as they construct and revise search statements. It is this combination of context and experience that constitutes the background knowledge of our experienced searcher.

For our first experiment, we did three runs (fsclt3, fsclt4 and fsclt4a). We manually constructed queries for all the topics. The searcher was permitted to employ any number of terms into simple or complex Boolean arguments. For this experiment, a single user query was entered for each topic, and a relevance ranked output was generated for each, using standard system features of the MPS Information Server. Once the queries were constructed, they were run producing a first set of results (fsclt3). We then allowed the searcher to review the ranked output and select up to two documents which would be applied as relevance feedback to the search (fsclt4). We then did a third run where the system automatically selected the first two documents from the initial set of results (fsclt3) and applied them as relevance feedback to the search to produce a new set of results (fsclt4a).

- What is the effectiveness of the document scoring algorithm when searching across multiple databases?

For our second experiment, the TREC-5 corpus was split and indexed into 100 separate databases as defined by the database merging track. We then took the manually constructed queries from the first experiment and ran them across the databases, merging the results into a single ranked list (fsclt3m). We then compared these results to the baseline run fsclt3.

The ranking algorithm was initially developed, tested and refined using the TREC-4 corpus and database merging track guidelines set out in TREC-4.

3 Searcher Model and Guidelines

Because all the runs employed the same query formulations, the same searcher model and guidelines apply to all of them. All query statements for the experiments were constructed by one person. The initial parameters of the searcher 'model' were defined as follows:

- s/he regularly searches on-line catalogs and bibliographic databases in an academic setting;

- s/he may have some search training, but is not a professional searcher;
- s/he dislikes reviewing large search outputs;
- s/he is seeking information to solve a real-life problem;
- s/he may not be a content expert in the topic area of a given question.

3.1 Instructions to Searcher

The following instructions guided query formulation:

- prepare a single search statement that will capture the most relevant documents for a given topic;
- use single or multiple terms, employing wild card capability to capture multiple versions of a word, and/or quotes around several words (e.g., "cardiac arrest") to create a fixed phrase;
- apply boolean logic as desired, using AND, OR or NOT operators. Create nested statements using parentheses if desired;
- consult the stop word list if needed, but no other databases are available for consultation;
- the total time taken to prepare a single query should not exceed 5 minutes.

3.2 Searcher Training

In preparation for the experiment, the searcher performed training exercises using the TREC-5 training data. First, general capabilities of the system and features of the search engine were described. Then, three topics were selected from the TREC-4 topics by the searcher. For each topic, a query formulation constructed by the searcher was run against the test database. Results were analyzed using the Trec Eval program. Lists of document headlines were provided to the searcher for examination. The searcher was allowed to reformulate and re-run training queries as many times as she desired. The search interface for training exercises employed a custom client application created for this experiment and MacWais, a freely available WAIS client for the Macintosh.

4 System Configuration

The MPS Information Server is a commercial full-text retrieval system that runs on a large number of Unix based platforms. Given a user query, the MPS system returns a list of relevance-ranked documents from a database. The system is capable of performing simple or complex term or phrase searching

using parentheses, wildcards and Boolean operators. Soundex, typographical variation (for example, missing letters, 'color' would also pick up 'colour', and juxtaposition of letters, 'animal' would also pick up 'ainmal') and fielded searching are also supported. The system is designed to favor precision over recall when performing searches. Because it supports a number of different protocols, including WAIS-88, Z39.50-V2, HTTP, CGI-BIN, Gopher+ as well as two internal protocols, the MPS Information Server is capable of responding to search requests from a wide variety of clients applications.

The TREC-5 experiments employed version 3.1 of the MPS Information Server running on a SparcStation 5/110. Four gigabytes of disk space were set aside and split evenly between data and indices. For the purposes of the experiments, we built a custom client application. Running on the SparcStation, the custom client communicated with the MPS Information Server using the WAIS-88 protocol. This client application was designed to read TREC topic files, build a query by extracting a specific field (or fields) from the individual topic entries, run the queries against the server and save the query results in the TREC result format to a specified file. The results files could then be processed by the Trec Eval program to obtain the precision-recall values for the run.

A special parser was built to index the TREC database. We created an artificial headline for each document by concatenating the document ID field (the <DOCNO>) and the document title (where a title was available). We then suppressed a number of fields judged to be 'noise' based on inspection of sample documents from the various sources (a complete list of the field suppressed is provided in appendix A). The rest of the documents were indexed as plain text, with the SGML tags extracted from the text, and the words stemmed using a plural stemmer. No additional information was extracted apart from the word positions to allow phrase and proximity searching if desired by the searcher. All keywords in the news articles were suppressed, as required by the instructions.

While the MPS system's indexer starts up with an initial stop-word list (containing 377 words), it can choose to convert a word to a stop word if that word's total occurrence in the database reaches a specified value. The stop word value, which is site- and collection-dependent, would typically be set anywhere in the range of 20,000 to 50,000 occurrences. For the TREC-5 experiments, it was set at 150,000 occurrences to retain as many words as possible in the database. This resulted in a final stop-word list of 446 words.

Two databases were created, one containing disks 2 and 3 (the ad-hoc training database) and the other containing disks 2 and 4 (the ad-hoc test database). Each database took approximately 8 hours to build; their index sizes were about 530 MB each (from an initial size of 1.9GB for the TREC-4 corpus, and 1.6GB for the TREC-5 corpus).

5 TREC-5 Results for FS Consulting Experiments

5.1 First Experiment

In the first experiment, the searcher initially created a single written query for each of 50 ad hoc topics. The queries were then run producing a first set of results (fsc1t3). The searcher was then allowed to review the ranked output and select up to two documents from the initial set of results which would be applied as relevance feedback to the search (fsc1t4). We then did a third run where the system automatically selected the first two documents from the initial set of results and applied them as relevance feedback to the search (fsc1t4a).

The relevance feedback algorithm employed for query expansion in this experiment works by ranking all terms in selected documents by frequency of occurrence. The ten most 'interesting' terms were chosen from that list for further use. The most 'interesting' terms were defined as neither the most frequent or infrequent terms. Rather, frequency parameters were specified to eliminate the high and low ends of the spectrum. The experiment's final result sets were produced by expanding each original query to include new terms, assigning weights to the old and new terms, and re-ordering documents based on new relevance weights.

This automated query expansion feature was designed as a tool that could be used by information seekers who, having retrieved a large set from an initial search, wish to increase the likelihood that all relevant documents retrieved were listed in the first 30 or 40 titles in the result set.

5.1.1 Searcher Performance

Training exercises influenced the searcher's query formulation behavior in the following ways:

- she preferred to use the wild-card capability to increase recall, rather than entering multiple forms of a word;
- she was reluctant to add long lists of multiple synonyms, believing that they would dilute search results;
- she tried work-arounds to avoid the stop-list for common words like 'states' and 'united' (e.g., using "United States" as a bounded phrase).

The following examples are typical formulations used for fsc1t3. The searcher wrote out the formulations, which were entered into the system by the researcher (FS) without further discussion or modification.

Topic 252: "illegal alien" OR "illegal immigra*"

Topic 259: ((John OR Jack) AND Kennedy) OR JFK) AND assassin*)

Topic 262: "seasonal affective disorder" OR SADS OR (seasonal AND disorder)

Topic 271: "solar power" OR "solar energy"

Most query formulations for fscit3 employed parentheses, wildcards and the AND and OR Boolean operators. As the examples indicate, not all capabilities of the system were employed (e.g., field searching, soundex, typographical variation and "NOT" operator were not used for example). The bounded phrase was the most common special feature used. Missing parentheses in the examples above suggest that logic statements were not always constructed properly.

5.1.2 Server Performance

The results for fscit3 produced the following precision/recall figures over all of the topics:

```
Queryid (Num):          50topics  fscit3
Total number of documents over all queries
Retrieved:              30987
Relevant:                5524
Rel_ret:                1866
Interpolated Recall - Precision Averages:
at 0.00                  0.5178
at 0.10                  0.3371
at 0.20                  0.2410
at 0.30                  0.1830
at 0.40                  0.1297
at 0.50                  0.1106
at 0.60                  0.0899
at 0.70                  0.0554
at 0.80                  0.0375
at 0.90                  0.0201
at 1.00                  0.0201
Average precision (non-interpolated) over all rel docs
0.1368
Precision:
At    5 docs:           0.2960
At   10 docs:           0.2720
At   15 docs:           0.2627
At   20 docs:           0.2510
At   30 docs:           0.2333
At  100 docs:           0.1584
At  200 docs:           0.1120
At   500 docs:          0.0631
At  1000 docs:          0.0373
R-Precision (precision after R (= num_rel for a query) docs
retrieved):
Exact:                  0.1870
```


Overall, for all topics, 34% of the relevant documents were retrieved from the database and only 6% of the documents retrieved were relevant.

The results for fsc1t4 produced the following precision/recall figures over all of the topics:

```
Queryid (Num):      50topics  fsc1t4
Total number of documents over all queries
  Retrieved:      50000
  Relevant:        5524
  Rel_ret:       2159
Interpolated Recall - Precision Averages:
  at 0.00      0.5757
  at 0.10      0.3632
  at 0.20      0.2824
  at 0.30      0.2147
  at 0.40      0.1599
  at 0.50      0.1297
  at 0.60      0.0918
  at 0.70      0.0660
  at 0.80      0.0385
  at 0.90      0.0241
  at 1.00      0.0187
Average precision (non-interpolated) over all rel docs
0.1559
Precision:
  At    5 docs:  0.3320
  At   10 docs:  0.3080
  At   15 docs:  0.2813
  At   20 docs:  0.2600
  At   30 docs:  0.2393
  At  100 docs:  0.1644
  At  200 docs:  0.1159
  At  500 docs:  0.0685
  At 1000 docs:  0.0432
R-Precision (precision after R (= num_rel for a query) docs
retrieved):
  Exact:      0.2006
```

Overall, for all topics, 39% of the relevant documents were retrieved from the database and just over 4% of the documents retrieved were relevant.

The results for fsc1t4m produced the following precision/recall figures over all of the topics:

```
Queryid (Num):      50
Total number of documents over all queries
  Retrieved:      50000
  Relevant:        5524
  Rel_ret:       2047
Interpolated Recall - Precision Averages:
  at 0.00      0.5439
  at 0.10      0.3605
  at 0.20      0.2502
  at 0.30      0.1812
```

at 0.40	0.1381
at 0.50	0.1130
at 0.60	0.0830
at 0.70	0.0561
at 0.80	0.0225
at 0.90	0.0194
at 1.00	0.0189

Average precision (non-interpolated) over all rel docs
0.1415

Precision:

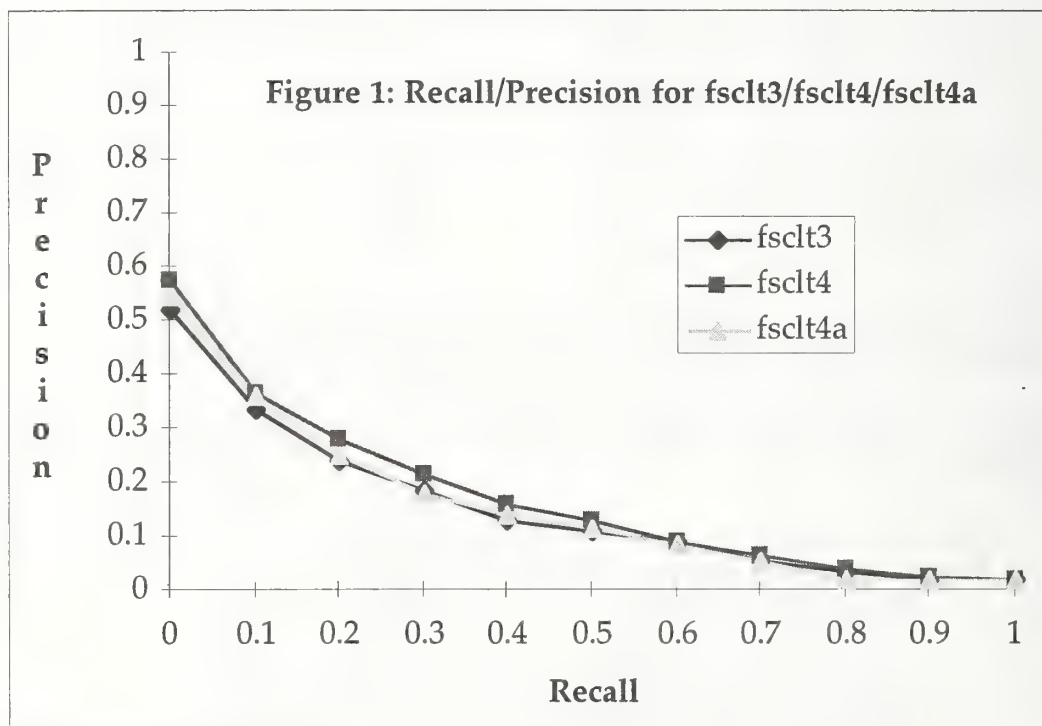
At 5 docs:	0.3120
At 10 docs:	0.3040
At 15 docs:	0.2787
At 20 docs:	0.2640
At 30 docs:	0.2400
At 100 docs:	0.1588
At 200 docs:	0.1101
At 500 docs:	0.0655
At 1000 docs:	0.0409

R-Precision (precision after R (= num_rel for a query) docs
retrieved):

Exact:	0.1993
--------	--------

Overall, for all topics, 37% of the relevant documents were retrieved from the database and just over 4% of the documents retrieved were relevant.

Figure 1 below show the precision-recall curve for both fsclt3, fsclt4 and fsclt4a.



These results would seem to suggest that using automatic relevance feedback would produce better results than not using it, but that letting the user choose which documents to apply as relevance feedback is still the best policy.

When comparing these results with our TREC-4 results [1], two differences jump out: the first one is that these runs returned a lot more documents overall as well as more relevant documents (better recall); the other is that the precision was higher in the TREC-4 results. This is in line with the design of the search engine which is to favor precision against recall, so when fewer documents are retrieved, precision increases.

5.2 Second Experiment

The second experiment was to measure the effectiveness of the document scoring algorithms when searching across multiple databases (database merging track). The TREC-5 corpus was split and indexed into 100 separate databases as defined by the database merging track. We then took the manually constructed queries from the first experiment and ran them across the databases, merging the results into a single ranked list (fsc1t3m). We then compared these results to the baseline run fsc1t3.

5.2.1 Server Performance

The results for fsc1t3m produced the following precision/recall figures for all the topics:

```
Queryid (Num):      all  fsc1t3m
Total number of documents over all queries
  Retrieved:      31189
  Relevant:        5524
  Rel_ret:       1389
Interpolated Recall - Precision Averages:
  at 0.00        0.6019
  at 0.10        0.3694
  at 0.20        0.2458
  at 0.30        0.1640
  at 0.40        0.1182
  at 0.50        0.0960
  at 0.60        0.0612
  at 0.70        0.0396
  at 0.80        0.0337
  at 0.90        0.0197
  at 1.00        0.0197
Average precision (non-interpolated) over all rel docs
0.1354
Precision:
  At    5 docs:   0.3720
  At   10 docs:   0.3240
  At   15 docs:   0.3013
  At   20 docs:   0.2760
```

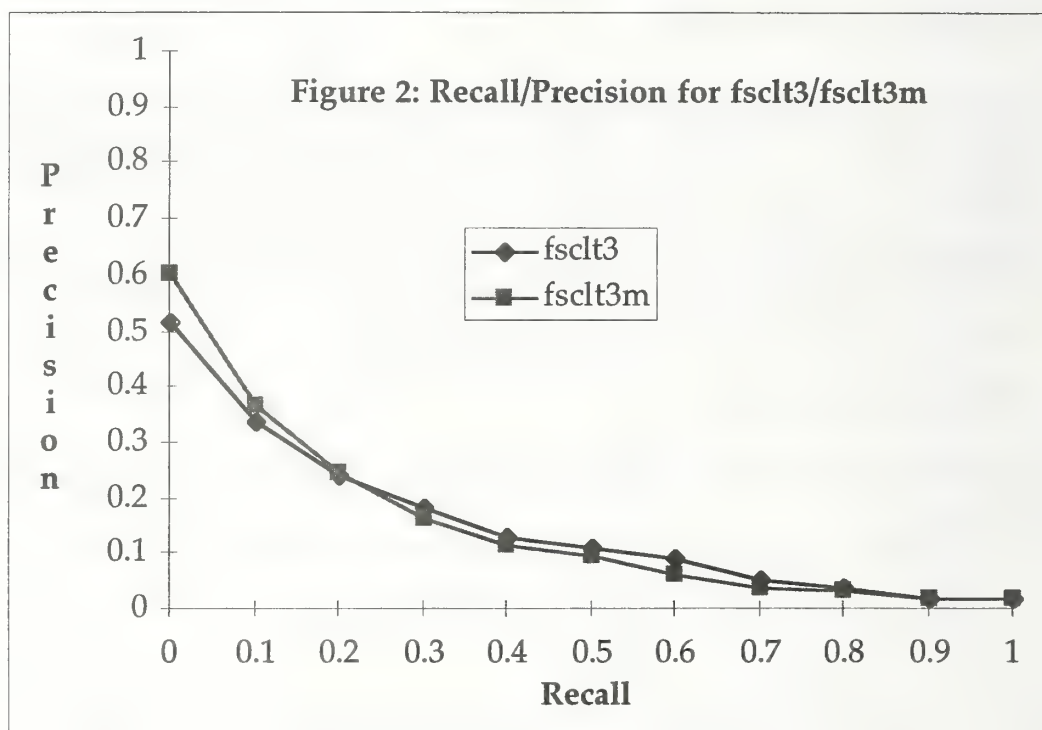

At 30 docs:	0.2467
At 100 docs:	0.1606
At 200 docs:	0.1063
At 500 docs:	0.0531
At 1000 docs:	0.0278

R-Precision (precision after R (= num_rel for a query) docs retrieved):

Exact:	0.1890
--------	--------

Overall, for all topics, 25% of the relevant documents were retrieved from the database and only 4.5% of the documents retrieved were relevant.

Figure 2 below show the precision-recall curve for fsclt3 and fsclt3m.



What is interesting to note is that while this run (fsclt3m) retrieved fewer relevant documents than fsclt3, precision is higher. This would seem to suggested that the document scoring algorithms works well when searching across databases.

While working on the document scoring algorithm we also ran the TREC-4 database merging track and got results very similar to those above whether the corpus was segmented into two databases or eight databases (as was required by the track). We repeated this experiment with the TREC-5 data and got the same results as well.

The implication of this is that we can segment a very large database across a number of machines to take advantage of parallelization (the MPS search engine can search multiple databases in parallel) and be able to present the user with a single, meaningfully ranked, results set. In addition one would also gain in terms of system redundancy where portions of the database would still be available for searching if one of the machines was down for repairs or maintenance.

6 Discussion of FS Consulting TREC-5 Results

The MPS search engine is designed to operate in an interactive setting, where quick response and high precision are generally preferable to high recall. (High recall can be achieved by creating several different queries for the same topic; this is the recommended search strategy when high recall searches are required). Comparing the TREC-5 results with our TREC-4 [1] results really illustrates this. While our TREC-4 results returned less documents overall, the precision was higher.

6.1 Searcher improvements

Examination of the query formulations indicated that the searcher did not take full advantage of system features, and constructed queries that were not optimal for the search system. For example, the wildcard feature in a lot of cases, thereby increasing recall at the expense of precision. Whether or not these formulations are typical of average searchers, it seems clear that the training period did not produce sufficient understanding of the system's strengths and weaknesses, nor were optimal query models presented and reinforced.

6.2 System improvements

While the relevance feedback algorithms work adequately at this point, it is hard not to want better performance from them. In that light we will be running a number of experiments this year prior to TREC-6 to fine-tune the relevance feedback algorithms further.

7 Future Work

TREC-5 experiments provided baseline results and in a non-interactive environment and allowed exploration of possible directions for future work. Several themes emerged that will guide our research efforts in preparation for participation in TREC-6.

- The system will be tuned and improved. The query expansion tool will continue to be tested and revised. Additional relevance feedback algorithms will also be tested. A user interface will be constructed to allow the searcher to review results and make other decisions about search parameters.
- Additional examples of manual query formulations will be gathered and tested for TREC-5 topics, in order to build and improve the model of an 'average' searcher. Efforts will be made to gather formulations from searchers with different backgrounds (e.g., librarians, medical students, academic faculty, administrative and clerical staff). These data will be used to improve searcher training, and to suggest additional user tools.
- An interactive experiment will be designed based on improvements and searcher modeling to be undertaken this year.

8 Acknowledgments

The author wishes to thank Valerie Florance for her invaluable help with the design of this experiment and in the creation of the queries.

9 References

[1] Schiettecatte, François and Florance, Valerie, "Document Retrieval Using the MPS Information Server". In Harman D. (Ed) The Fourth Text Retrieval Conference (TREC-4). National Institute of Standards and Technology Special Publication 500-236, Gaithersburg, Md. 20899.

Appendix A

The following fields were suppressed at indexing time:

AP Newswire:	<PAGE>	<APD>
<DOCNO>		<ISD>
<FILEID>	Wall Street Journal:	
<FIRST>	<DOCNO>	SanJose Mercury
<SECOND>	<DD>	<DOCNO>
<DATELINE>	<SO>	<ACCESS>
<BYLINE>	<IN>	<CAPTION>
<NOTE>	<DATELINE>	<DESCRIPT>
<UNK>	<G>	<SECTION>
	<GV>	<MEMO>
Congressional Record:	<AN>	<BYLINE>
<DOCNO>	<RE>	<COUNTRY>
<DOCID>	<MS>	<CITY>
<CENTER>	<NS>	<EDITION>
<DATE>	<DATE>	<CODE>
<FLD001>	<DO>	<NAME>
<FLD002>	<ST>	<PUBDATE>
<FLD003>		<DAY>
<SO>	Ziff:	<MONTH>
	<DOCNO>	<PG.COL>
Federal Register 88:	<DOCID>	<PUBYEAR>
<DOCNO>	<DESCRIPT>	<REGION>
<DOCID>	<JOURNAL>	<STATE>
	<AUTHOR>	<WORD.CT>
Federal Register 94:		<DATELINE>
<DOCNO>	Patents:	<COPYRGHT>
	<DOCNO>	<LIMLEN>
Financial Times:	<WKU>	<LANGUAGE>
<DOCNO>	<SRC>	
<PROFILE>	<APN>	
<DATE>	<APT>	
<PUB>	<ART>	



Using Relevance Feedback

within the Relational Model for TREC-5

David A. Grossman
Office of Information Technology
3E09 Plaza B
Washington, DC 20505
dgrossm1@mason1.gmu.edu

Carol Lundquist
George Mason University
Fairfax, VA 22091
clundqui@osf1.gmu.edu

John Reichart
P2000 Technology Inc.
P.O. Box 916
Hanover, NH 03755
jreicher@lccinc.com

David Holmes
NCR
2 Choke Cherry Drive
Rockville, MD 20850
holmed@uf4725p02.WashingtonDC.ncr.com

Abdur Chowdhury
George Mason University
Fairfax, VA 22091
achowdhu@gmu.edu

Ophir Frieder*
Department of Computer Science
Florida Institute of Technology
ophir@cs.fit.edu

Abstract

For TREC-5, we enhanced our existing prototype that implements relevance ranking using the AT&T DBC-1012 Model 4 parallel database machine to include relevance feedback. We

identified SQL to compute relevance feedback and ran several experiments to identify good cutoffs for the number of documents that should be assumed to be relevant and the number of terms to

add to a query. We also tried to find an optimal weighting scheme such that terms added by relevance feedback are weighted differently from those in the original query.

We implemented relevance feedback in our special purpose IR prototype. Additionally, we used relevance feedback as a part of our submissions for English, Spanish, Chinese and corrupted data. Finally, we were a participant in the large data track as well. We used a text merging approach whereby a single Pentium processor was able to implement adhoc retrieval on a 4GB text collection.

* This work supported in part by the National Science Foundation under contract number IRI-9357785 and industrial matching funds under the National Young Investigator Program. Ophir Frieder is currently on leave from the Department of Computer Science at George Mason University.

1. Introduction

For TREC-5, we implemented relevance ranking queries using SQL on an AT&T DBC-1012 (formerly Teradata) parallel database machine [1]. This was an extension to our prior work which implemented the vector-space model as an application of a relational DBMS. Additionally, we implemented a special purpose IR prototype to test a new index compression algorithm and to provide performance comparisons to the relational approach.

We submitted official results for the 2GB English collection, both for automatic and manual adhoc queries and against the Spanish and Chinese collections. We also submitted results using n-grams to process the corrupted data. Each of these submissions included relevance feedback.

We briefly describe the implementation of relevance feedback in our relational prototype and our special-purpose prototype in Section 2. More detailed descriptions are found in [2, 3]. Sections 3, 4, and 5 will describe the results obtained for our English, Spanish, and Chinese submissions. Section 6 describes our corrupted data results. Our conclusions are outlined in Section 7.

2. Implementation of Relevance Feedback

We developed two separate implementations, a parallel relational approach and a special purpose IR approach.

2.1 Implementation on the DBC

Our approach treats the information retrieval (IR) problem as an application of a relational database system. While parallel implementations of relational database systems are common, parallel implementations of IR systems are rare. Work done on large scale relevance feedback did include a parallel machine, but this work did not include within document frequencies. We implemented relevance feedback as an extension of the vector space model with standard *tf-idf* weights.

We model an inverted index with a relation `DOC_TERM(doc_id, term, tf)`. A relation, `QUERY(query, term, tf)` indicates the terms and their frequency in the query. `DOC(doc_id, doc_name, doc_weight)` contains the document name and the normalized weight for each document. `QUERY_WEIGHT(query, query_weight)` contains the normalized query weight for each query. Finally, the `IDF(term, idf)` relation stores the inverse document frequency for each term.

Given these relations, the following SQL computes a cosine similarity coefficient for a given query: *query_number*.

```

Ex: 1  SELECT a.query, c.doc_name, SUM(a.tf * b.tf * e.idf * e.idf)/
      Sqrt(d.query_weight * c.doc_weight)
FROM QUERY a, DOC_TERM b, DOC c, QUERY_WEIGHT d, IDF e
WHERE a.term = b.term
      AND a.term = e.term
      AND b.docid = c.docid
      AND a.query = d.query
      AND a.query = query_number
GROUP BY a.query, c.docname, d.query_weight, c.doc_weight
ORDER BY 3 DESC;

```

Assume the query given above has been executed, and the top n document identifiers are stored in the relation *TOP_DOC(doc_id)*. To compute relevance feedback, the top t terms (sorted by some sort criteria) found in the top n documents are added to the query. This is accomplished with standard SQL used in each of the following steps:

- Step 1 - Identify the top n documents for each query through relevance ranking.
- Step 2 - Identify the terms from the top n documents.
- Step 3 - Select the feedback terms to be used for relevance feedback.
- Step 4 - Merge the feedback terms with the original query.
- Step 5 - Identify the top documents for the modified queries through relevance ranking.

Each step is implemented by a standard SQL statement. Although a single SQL statement could be implemented, for clarity we use separate SQL statements. Hence, it is relatively straightforward to extend the relational approach to include relevance feedback.

2.2 Special Purpose IR Prototype

We also extended our special-purpose IR system to include relevance feedback. Our system implements relevance ranking using the vector-space model with the cosine similarity measure using *tf-idf* weights [4]. Implementation of relevance feedback was done by obtaining the top n documents and parsing their original text to find the terms in these documents. The terms were then sorted according to a specified sort order and added to the original query.

3. English Results

3.1 Automatic

We submitted both manual and automatic results for the adhoc collection. Each section of the corpus was loaded into a corresponding relation, and a larger query to UNION all the different relations was implemented. In addition to simply loading terms, we also loaded phrases which were recognized with a crude phrase parser. A phrase was defined as a two term sequence that did not contain a punctuation mark or a stop word. The topics were parsed in the same fashion and both terms and phrases were incorporated into the queries. Phrase inverse document frequency (IDF) was computed as if the phrase was a single term. All terms other than stop words were used in the query.

Our first submission, *gmu96au1* used our relational prototype. Only terms from the <desc> portion (i.e., short version) of the query were used. Terms from the top 10 documents

for the original query were identified. These were sorted by $n*idf$ as given in [5] where n is the number of top ranked documents that have the term (n is between 1 and 10). The top 10 terms were added to the original query, duplicates were removed, and the query was executed again. Only a single iteration of relevance feedback was used.

The second submission, gmu96au2, used our special purpose IR prototype. Terms from both the <desc> and the <title> components of the query (i.e., long versions of the query) were used. The cosine similarity measure was executed for these terms, and again, the top 10 documents were assumed to be relevant. These terms were sorted by the same $n*idf$ measure; however, the top 20 terms were added to the original query. These terms were added to the original query, and the cosine measure was computed. A scaling factor of .4 was applied to the new terms (several scales were tested on the TREC-4 collection).

3.2 Manual

The key difference in our two manual adhoc submissions is the use of manually assigned weights versus automatically assigned weights.

3.3 Manually Assigned Weights

Our first manual submission, gmu96ma1, used manually assigned terms and manually assigned weights. The terms for each query were derived by examining the initial query and identifying terms and phrases that appeared relevant. Since the document collection was not stemmed, many variants based on prefixes and suffixes are included. Relevance feedback was also used. Queries were executed using manual term selection and terms in the top ranked documents that appeared to be of potential benefit were then added to the query. Subsequently, a new query based on this manual feedback was executed and our final run used the results from this query.

The assumption is that queries are about one or more concepts. Terms are grouped into a "concept" via the operator given below. Up to three concepts are supported, hence an operator of 1, 2, or 3 indicates the term is in a particular concept. For a document to be ranked, it must have at least one term in each concept (unless the term is placed in a special concept 0 – in this case the document may not have the term and still be ranked). Once this condition is satisfied, all other terms are used to contribute to the similarity coefficient. The similarity coefficient is computed as the sum of the manually assigned weights for which a match occurs. This score is then divided by the total number of terms and phrases in a document (not including stop words). Negative weights were assigned for query terms that were specifically excluded relevant documents (i.e., "find info about taxes worldwide, NOT in the US")

3.4 Automatically assigned weights

In our second manual run, gmu96ma2, the basic approach was similar to the first run. All terms remain the same, but the term weights and ranking algorithm differ. The term weights that were used were automatically computed as the *idf* (inverse document frequency). The ranking algorithm still used the three concept sets. The similarity coefficient was computed using the cosine similarity coefficient. However, normalization was done based on the total number of non-stop words in the document rather than the typical cosine length normalization.

3.5 Results

Our overall results for English are given below:

Test Run	Description	Avg. Precision	Above Median	Below Median	Equal Median
gmu96au1	Automatic (Relational IR, relevance feedback with top 10 terms)	.1079	10	37	3
gmu96au2	Automatic (Special IR, relevance feedback with top 20 terms, .4 scaling factor for new terms)	.1331	13	35	2
gmu96ma1	Manual (manually assigned weights)	.2147	21	27	2
gmu96ma2	Manual (automatic assigned weights)	.2141	19	26	5

It is reasonable to expect that our two automatic implementations would have similar results as their basic techniques are nearly identical. Our calibrations showed that a scaling factor did slightly improve precision/recall, and that is verified here. It should be noted that our calibrations consistently found precision/recall in the .20 to .22 range on the TREC-4 collection. We are currently investigating the reason for this reduction in precision/recall when the same approach was used on the TREC-5 collection. The manually assigned weights performed no better than automatically assigned weights for the manually constructed queries.

3.6 Failure Analysis

One of the interesting features of relevance feedback is that while relevance feedback improves precision/recall for some queries, it also decreases precision/recall for others. It would be useful to be able to predict those queries which would benefit from relevance feedback so that relevance feedback would not be applied to those queries whose precision/recall would decrease. In an effort to identify such a predictor, we analyzed the query terms both before and after relevance feedback for the gmu96au1 run. Based on this analysis, the queries were divided into three groups: queries that relevance feedback improved precision/recall (16 queries), queries that relevance feedback did not change precision/recall (14 queries), and queries that relevance feedback decreased precision/recall (20 queries).

TOPICS	AVG # TERMS	AVG IDF OF TERMS Original Query			AVG IDF OF TERMS After RF		
	PER QUERY	IDF	MAX IDF	MIN IDF	IDF	MAX IDF	MIN IDF
IMPROVED BY RF 254,257,258,259,264,265,267,273,280,282, 283,284,287,288,298,299	15.1	2.46	0.90	4.80	2.96	1.63	4.43
UNCHANGED BY RF 252,253,256,260,263,268,272,278,279,281, 292,296,297,300	16.3	2.38	0.82	4.73	2.61	1.32	4.30
DECREASED BY RF 251,255,261,262,266,269,270,271,274,275, 276,277,285,286,289,290,291,293,294,295	16.6	2.41	0.90	4.83	2.74	1.43	4.43
ALL QUERIES	16.0	2.41	0.88	4.79	2.77	1.46	4.39

The table below illustrates the terms both before and after relevance feedback for several queries.

TOPIC	IMPACT OF RELEVANCE FEEDBACK	TERMS AND PHRASES	
		Original Query	New Terms Identified by RF
#264 Identify instances where U.S. citizens have been or are being held in foreign jails since the year 1900.	Improved 320%	<ul style="list-style-type: none"> • jails since • foreign jails • identify instances • jails • being held • instances • identify • citizens • held • foreign • year • being • since 	<ul style="list-style-type: none"> • longest held • nine americans • chief middle • south lebanon • prisoners • release
# 272 Medically, is outpatient surgery more prevalent now than ever before?	No Change	<ul style="list-style-type: none"> • surgery more • outpatient surgery • medically • outpatient • surgery • ever • more 	<ul style="list-style-type: none"> • surgery centers • inpatient • surgical • health care • hospital • medical • care

<p># 262</p> <p>Is seasonal affective disorder syndrome (SADS) (also known as seasonal absence of daylight syndrome), a worldwide disorder?</p>	<p>Decreased 100%</p>	<ul style="list-style-type: none"> • sads • seasonal affective • affective disorder • affective • daylight • disorder • syndrome • seasons • worldwide • absence • known 	<ul style="list-style-type: none"> • phobias • tics • disorders • symptoms • depression • illness • disease
--	-----------------------	---	---

As seen in the table above, the average *idf* of terms in the queries is very similar for those queries that were improved by relevance feedback to those queries that were not improved. However, after relevance feedback was applied, queries which benefited from relevance feedback had a slightly higher average *idf* than queries whose effectiveness was decreased by relevance feedback. The relationship between the weight of the terms in a query and the improvement obtained from relevance feedback needs further investigation.

4. Spanish Results

For the Spanish data, we used the relational prototype to obtain our automatic results. Both results were done with relevance feedback with only a difference in scale between each result.

We developed a Spanish stop word list by identifying the top 500 most frequent terms and asking a Spanish linguist to determine which ones were really not so common across the language that they should be in a stop list.

Essentially the same approach was used as during our adhoc run. The top ten documents were identified by using Spanish terms and the usual cosine measure. The terms in these documents were ordered by the $n*idf$, and the top ten terms were added to the query. The initial run does not do any scaling while the second run increases all phrase weights in the query by a factor of five.

4.1 Results

Our results for Spanish data are given below:

Test Run	Description	Avg. Precision	Above Median	Below Median	Equal Median
	Cosine	.2215	6	19	0
gmu96sp1	Cosine+rf	.2403	6	18	0
gmu96sp2	Cosine+rf+scale	.1900	4	21	1

The baseline run was not submitted as an official result, but it is given here to measure the effect of relevance feedback. Relevance feedback without scaling improved precision by a small margin (around 8%). The use of scaling **clearly did not improve** performance for these queries.

5. Chinese Results

For the Chinese data, we used our special purpose IR prototype. We implemented both manual and automatic relevance feedback.

5.1 Automatic

The first run is a baseline run. The cosine measure was used with tf-idf weights. To parse Chinese, we took the simplistic approach of assuming each term was one two-byte character. No stop words were used. We used all Chinese components of the query (description, narrative, and title). The second run using automatic relevance feedback with the same technique as described for adhoc English. Without any training data we were forced to assume that Chinese would perform in a similar fashion to English for relevance feedback.

The results from the first run were obtained, and the top 10 documents for each query were identified. The top 20 terms were selected based on the $n*idf$ measure. The original query was then augmented to use the new terms, and a scaling factor of .4 was applied to the new terms. These values were obtained from calibration using the English data with TREC-4 qrels and mirror one of our English submissions.

5.2 Manual

Both of our manual runs use manual relevance feedback. Instead of blindly assuming that the top 10 documents were relevant, we asked two people who were fluent in Chinese to read the top ten documents and indicate which ones were relevant. Once this was done the top ten terms from these documents were added to the collection, and the same computation as done for the automatic runs was computed. Two differences exist between the two manual runs. The first is that a different relevance assessor was used for each run. The second is that the entire query is used in run 1, but only the <description> portion of the query is used in run 2.

5.3 Results

Our results for Chinese data are given below:

Test Run	Description	Avg. Precision	Above Median	Below Median	Equal Median
gmu96ca1	Automatic Cosine	.2955	8	10	1
gmu96ca2	Automatic Cosine+rf	.3274	12	6	1
gmu96cm1	Manual Cosine+rf+whole query	.3279	12	7	0
gmu96cm2	Manual Cosine+description	.3065	11	8	0

The results indicate that relevance feedback is of benefit for Chinese data. Also, we again find that the manual effort (in this case reading nearly 200 pages of printed Chinese) did not yield any significant improvement over the automatic approach.

6. Corrupted Data Results

With corrupted data, we relied upon 4-grams (overlapping sequences of four characters) to be resilient to errors in text. Our first submission used a standard cosine measure on all three test collections (baseline, 5% corrupted, and 20% corrupted). Our second submission incorporated relevance feedback using n-grams.

As the test collection for the confusion track did not contain data for which relevance assessments existed, it was not possible to calibrate for the data on this collection. Hence, we used the exact same relevance feedback technique that we used for the adhoc English collection except that terms are replaced by n-grams. The standard cosine measure was used with the exception that terms were replaced with 4-grams. The n-grams spanned term boundaries. Hence, for an input phrase of New York, the n-grams were: new_, ew_y, w_yo, york, ork_.

The query was generated by parsing the input query and generating its component n-grams. We used the same stop word list as used for the adhoc English collection. Any term that was found on this list was eliminated before n-grams were generated. In addition to this, we generated a stop-n-gram list in which the top .05% n-grams were eliminated. The n-grams were sorted based on their collection frequency. Also, we ensured that no more than 150 n-grams were added to this list. The stop-n-gram list was generated for each version of the corrupted data (baseline, 5% corrupted, and 20% corrupted). We used the same relevance feedback process as used for the adhoc English collection. N-grams were parsed and a cosine measure was computed using n-grams instead of terms. The top n-grams in the top ten documents were identified and were sorted by the same $n \cdot idf$ measure. The top 20 n-grams were used with a scaling coefficient of .4.

Our results for corrupted data are given below. Since this was a search for a known item, we give the mean of the reciprocal of the rank at which the known item was found for all 49 queries.

Test Run	Description	Degrade 0	Degrade 5%	Degrade 20%
gmu961	Cosine	.39	.31	.22
gmu962	Cosine+rf	.20	.19	.15

7. Conclusions and Future Work

Given that this was only our second year as a Category A participant, we still see much room for improvement. Overall, we confirmed a known result that relevance feedback is clearly of benefit to English language processing. Our new work in this area is that relevance feedback can be implemented using the relational model. This yields a portable, parallel approach to computing relevance feedback. We experimented with several sort orders to find the optimal number of documents to retrieve and number of terms to add to the query and we ended up at 10 documents retrieved with either 10 or 20 terms to add to the query. As work done in [6] indicates that other term weighting methods outperform the *tf-idf* weights, we will incorporate this information and develop relational implementations using standard SQL of alternative term weighting methods. Also, we will continue the search for indicators of when relevance feedback should be applied and when it should not be applied to individual queries.

The use of relevance feedback clearly helped our manual queries as well. Our manual English results were more than double the precision of our automatic approach. However, a significant amount of time per query (15 to 30 minutes) was used to develop these queries. Relevance feedback worked reasonably well with Chinese data and we have an initial result that suggests that manual relevance feedback does not improve on automatic relevance feedback.

Overall, our final numerical results were similar to our results for TREC-4. Our calibration during the year suggested that our effectiveness would increase by 10 to 20 percent, but we were unable to calibrate with Chinese or corrupted data. We will use the collection from TREC-5 as training data for future work.

References

- [1] AT&T Global Information Systems. *Teradata DB2-1012 Concepts and Facilities*, March 1992.
- [2] D. Grossman, O. Frieder, D. Holmes, and D. Roberts. Integrating Structured Data and Text: A Relational Approach, *Journal of the American Society of Information Science*, January 1997.
- [3] D. Grossman, Integrating Structured Data and Text: A Relational Approach, Thesis, George Mason University, 1996.
- [4] G. Salton, C.S. Yang, and A. Wong. A vector-space model for information retrieval. *Communications of the ACM*, 18, 1975.
- [5] D. Harman. "Relevance Feedback Revisited," *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Ed. Nicholas Belkin, Peter Ingwersen and Annelise Mark Pejtersen, SIGIR Forum, June 21-24, 1992.
- [6] C. Buckley, A. Singhal, M. Mitra, and G. Salton, "New Retrieval Approaches Using SMART: TREC 4," Text Retrieval Conference, sponsored by National Institute of Standards and Technology and Advanced Research Projects Agency, November 1995.

TREC-5 Ad Hoc Retrieval Using K Nearest-Neighbors Re-Scoring

Ernest P. Chan, Santiago Garcia and Salim Roukos

<epchan|sgarcia|roukos>@watson.ibm.com

IBM T. J. Watson Research Center,

POB 718

Yorktown Heights, NY 10598

April 22, 1997

1 Introduction

In our first participation in TREC, we focus on improving on baseline results obtained from another search engine by means of automatic query expansion. We call the specific formula we used for query expansion “Knn re-scoring”, where “Knn” stands for “K nearest-neighbors”. The first-pass ranking is done using Okapi system’s basic scoring formula[1]. The documents are then rescored using the same formula with the top-ranked K documents as queries, weighted according to their first-pass scores. As we shall see in Sec. 5 below, the formula is motivated by viewing the rescoring process as a Markov process. This approach improves the precision substantially outside the topK retrieved documents.

We have tested a variety of other techniques in trying to improving the system. These include word-sense disambiguation, passage retrieval, and document length suppression. Although they do not yield substantial or consistent improvements, some insights into search techniques can nevertheless be extracted.

Our experiments are done using the short version of the ad-hoc TREC-5 queries with just the description field retained. The official entry is submitted

	disk2
ap	HEAD, TEXT
fr	TEXT
wsj	HL, LP, TEXT
ziff	ABSTRACT, SUMMARY, TEXT
	disk4
cr efiles	CENTER, H2, SO, TEXT, TI, TTL, UL
cr hfiles	H2, SO, TEXT, TI, TTL, UL
fr94	ACTION, ADDRESS, AGENCY, DOCTITLE, FOOTNOTE, FURTHER, SIGNJOB, SUPPLEM, TABLE, TEXT, USBUREAU, USDEPT
ft	CN, CO, HEADLINE, IN, PE, TEXT , TP

Table 1: Fields retained in TREC-5 documents

as *ibms96a*. For comparison purposes, performance on TREC-4 data and other smaller corpora are also reported here.

2 Preprocessing, Morphological analysis and Sensing

As mentioned in the Introduction, we utilize only the description field of the queries, since we anticipate users of our system will not enter more than one sentence. For the candidate documents, we discard a number of fields which contain noisy or useless content. A list of the retained fields can be found in Table 1.

The words in both the queries and the documents are stemmed using a morphological analyzer such that only the root of a word remains. The morphological analyzer also tokenizes the text such that punctuations are stripped, and common bigrams are joined together as single tokens (e.g. “because of” becomes “because_of”). This analyzer is built upon a part-of-speech tagger [3]. Once the words are tagged, a table lookup enables one to extract the root of any morphs.

We have previously studied the efficacy of the morphological analyzer and

	AveP	P5	P10	P30	P100
words	0.2488	0.4800	0.4460	0.3513	0.2274
cased morphs	0.3319	0.5920	0.5340	0.4127	0.2766
uncased morphs	0.3507	0.5920	0.5540	0.4373	0.2910
manual cased morphs	0.3560	0.6200	0.5540	0.4367	0.2914

Table 2: Effects of morphological analysis and true-casing. First-pass (Okapi with unigrams) results on WSJ portion of TREC-3

	AveP	P5	P10	P30	P100
words	0.5800	0.3040	0.1960	0.0760	0.0264
cased morphs	0.8083	0.3520	0.2240	0.0867	0.0264

Table 3: Similar to Table 2: Results on radio broadcasts.

its true-casing facility on the WSJ¹ portion of TREC-3 data (see Table 2), and also on a small database of 140 radio broadcasts stories² (see Table 3). We use the Okapi ranking formula for this (see Section 4.) While we found that the analyzer does help retrieval accuracy, true-casing does not. This is because the true-caser does not have sufficient accuracy. The capitalization of some non-proper nouns in the TREC-3 queries are particularly problematic. We have manually removed these capitalization in the queries, and re-run the ranking program. In this case, cased morphs do perform slightly better than non-cased morphs (see last row in Table 2). In producing the official TREC-5 results, we use only uncased morphs.

We have also studied the effects of word-sense disambiguation[4] on the retrieval accuracy, although we have not incorporated it into the official system. Scores resulting from sensed queries³ are combined linearly with scores from unsensed queries (which include both unigram and bigram features, as explained in Sec.3) in a 9 to 1 ratio. The effects on TREC-4 results are quite obvious: even though the average precision with sensed words is lower, the precision at top10 is higher.(See Table 4.) This is an effect that we will observe again and again: the more specific a query, the higher the precision at

¹The queries in this case are the TREC-3 ad-hoc queries.

²We made up some 80 short queries with about 6 words each.

³Note that only a selected number of words in each query are sensed.

the top few scores, at the expense of a lower precision at the tail end of the ranked list. The opposite effects happen when we expand a query. We will elaborate on this point in Section 9. We note that the effects of sensing on TREC-5 is less encouraging. Top5 precision did not improve although top10 does. (See Table 5.) This may be because far fewer words in the TREC-5 queries were sensed. We will examine this in detail in [4].

Before extracting the term frequency information from the queries and documents, we filter them through a standard stopword list of 940 words.

3 Bigrams extraction

We extract all pairs of words from the text that are within a running 4-word window and form bigrams. (Stopwords do not count in this window.) To economize on storage space, however, only bigrams with both words in the queries vocabulary are retained. The bigrams are not ordered: i.e. (w_1, w_2) is the same as (w_2, w_1) .

4 First-Pass Ranking

For first-pass ranking, we adopted the basic Okapi [1] formula from TREC-3. Using their notation, each (non-stop) word in the query and document contributes a weight of

$$w = \frac{tf}{0.5 + 1.5 \times \frac{dl}{avdl} + tf} \times w^{(1)} \times qtf, \quad (1)$$

where

$$w^{(1)} = \log\left(\frac{N - n + 0.5}{n + 0.5}\right),$$

is the usual inverse document frequency (*idf*) factor. We can use the same scoring formula for the bigrams also, with two minor qualifications: (1) $w^{(1)}$ is set to 1.0 and (2) dl and $avdl$ refer to the document lengths in bytes of the original text, not the number of bigrams in that text. After the official evaluation, we have tried an experiment where the bigrams have true $w^{(1)}$ obtained from the corpus. We found indeed that there is a small benefit in using *idf*'s in bigrams scoring.

The unigram and bigram scores from Eq.(1) are combined linearly such that the unigram scores have a weight of 0.6 versus 0.4 for the bigrams. The result of these experiments are reported in Table 4 and Table 5. It is clear that the greatest benefit of adding the bigrams is in improving the precision at top5 retrieved documents.

5 Knn Re-Scoring

In order to expand the original query, we select the topK ($K = 10$ in our runs) documents from the first pass and use these as queries to re-rank the topN documents ($N = 1000$). Let $P(d_k|q)$ be the first-pass Okapi score of the k-th document, and $P(d_i|d_k)$ be the Okapi score of the i-th document using the k-th document as query. Then we define the second-pass score of document i to be

$$s_i = \sum_{k=1}^K P(d_i|d_k)P(d_k|q). \quad (2)$$

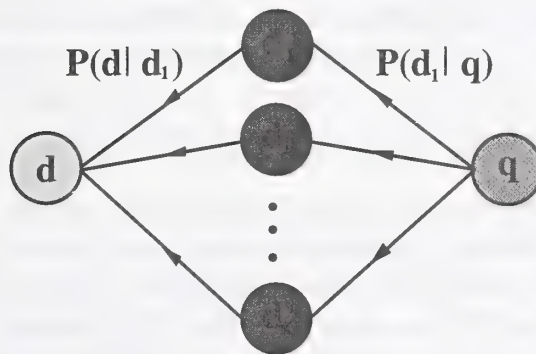


Figure 1: Knn Re-scoring

A graphical representation of this is shown in Figure 1. The notation is designed so that the formula resembles that of a Markov model, which was the original motivation for it.

We can just use the words in the expanded query k to compute s_{ik} , or we can include the bigrams in k . As mentioned above, the bigrams are restricted to those with words in the original (first-pass) query vocabulary.

Again, the unigram and bigram scores from Eq.(2) are combined linearly such that the unigram scores have a weight of 0.05 versus 0.95 for the bigrams. The addition of bigrams improves the second-pass scores slightly. (See Table 5.)

The overall improvement due to Knn re-scoring has been modest. In fact, there is a decrease in precision at the top10 documents. Our previous experience with Knn re-scoring in TREC-4 and other corpora has been much more positive. In Sec. 9, we shall discuss a possible cause of this.

6 Passage Retrieval

One may hypothesize that some documents may be viewed as relevant to a topic simply because of one particular passage within it, while other passages discuss irrelevant matters. If this view is correct, segmenting the documents into passages and scoring each of them separately, taking the maximum passage score as representative of the whole document may improve retrieval performance.

We have performed this experiment on TREC-4 data by segmenting documents into overlapping passages of about 100 words. The overlap is about 50 words. The result of using the first-pass "best passage" score is substantially worse than taking the document as a whole (with aveP=0.1803 vs. 0.1980 without passages), as can be seen from Table 4, seemingly disprove the hypothesis.

We have also tried using passage retrieval on the second pass. This can be done in 2 ways: using passages in the expanded queries to re-score the documents, and/or scoring the best passage in the document. We have tried various combinations of these 2 ways on the TREC-4 data. Furthermore, we can have a weighted average of first-pass passages used as expanded queries using the following weighting formula:

$$\begin{aligned} w_b &= 1/n^\xi \\ w_o &= \frac{1-1/n^\xi}{n-1}, \end{aligned} \tag{3}$$

where w_b is the weight of the best first-pass passage, and w_o is the weight

of the other passages, n is the number of non-overlapping passages⁴, and ξ ($0 \leq \xi \leq 1$) is the parameter that let us control the weights distribution. If $\xi = 0$, we use only the best passage as expanded query; if $\xi = 1$, all passages contribute equally, and we are back to using the whole first-pass document as expanded query. We again find that the best combination is obtained when we score the whole document (not its passages). In the future, one can combine the best-passage score and the whole-document score to see if this is better than each alone. Whether we use weighted passages as queries does not seem to have much effect.

Because of this discouraging result, we did not attempt passage retrieval in TREC-5.

7 Length Suppression

We believe that long documents have an unfair advantage over short ones because they often have a variety of heterogeneous subtopics which increases the chance of matching the query words. However, the occurrence of these subtopics does not necessarily make the document relevant to the query. Hence we have applied a sigmoidal suppression factor for document length, such that the new 2nd pass score s_{new} is related to the old 2nd pass score s_{old} by

$$s_{new} = \frac{1}{1 + e^{(dl - \text{mean}Dl)/width}} s_{old},$$

where $\text{mean}Dl = 10000$ and $width = 8000$. Scores of documents with length much larger than $\text{mean}Dl$ will be exponentially decreased.

It turns out that length suppression hurts precision for the top5 documents but is beneficial beyond that. In the future, we will adopt the technique advanced by the SMART group [5] to discover the bias (with respect to document length) that our system suffers.

⁴If we happen to pick a best passage that is offset by an odd-multiple of 50 words, we would simply ignore either the first 50 words or the last 50 words of the document for simplicity.

8 Combining 1st and 2nd pass Scores

Motivated by the idea of using a “prior probability” in a Bayesian probabilistic retrieval framework, we can combine the second-pass scores linearly with the first-pass scores, thus viewing the first-pass scores as the log of a prior probability. We use a (unnormalized) weight 0.01 for the first-pass vs. 0.99 for the second-pass scores. This generally produces a small improvement in both TREC-4 and TREC-5. The results are listed also in Table 4 and Table 5. The larger improvement in TREC-5 is expected because, as we discuss in the Conclusion, the initial retrieval there is so poor that using these documents for query expansion often hurts precision. Adding the first-pass scores ensures that movement in document ranks is reduced.

A more radical method to ensure that Knn re-scoring does not degrade the topK precision is to fix the rankings of the top10 or so documents from first pass and simply re-score the rest in the second pass. The results of not re-scoring the top10 in second pass are shown in Table 4 and Table 5. In TREC-5, it does show a substantial improvement over the official second pass score; however, this result does not hold for TREC-4. Hence it is still possible that Knn re-scoring not only improves the aveP, but can improve the topK precision as well.

9 Conclusion

We have implemented a query expansion scheme using all the words and some of the bigrams in the topK documents. An improvement in average precision is achieved by this scheme. When we tested this scheme on TREC-4 topics, the percentage improvement in aveP is more than 9 times bigger than the TREC-5 improvement (see Table[4]). The unimpressive performance in TREC-5 may be due to the fact that the initial retrieval has so poor accuracy that 2/3 of the top10 documents used as expanded queries are irrelevant. Nevertheless, we achieved an above-median average precision in 34 out of 50 topics compared with other TREC-5 (short queries) participants.

Compared to other methods of query expansion, such as that used by the SMART TREC-4 engine [2], we have used far more words and bigrams for expansion. Apparently this has hurt our performance, perhaps by overly generalizing the very short queries. In particular, the topK precision in

1st pass	2nd pass	AveP	P5	P10	P30	P100
u	–	0.1980	0.5240	0.4660	0.3447	0.2314
u, b	–	0.2014	0.5160	0.4440	0.3487	0.2356
u, b-idf	–	–	–	–	–	–
u, b, s	–	0.1987	0.5320	0.4740	0.3540	0.2310
u, b	u	0.2150	0.5120	0.4800	0.3960	0.2664
u, b	b	0.1629	0.4000	0.3600	0.2800	0.1938
u, b	b-idf	0.1678	0.4280	0.3700	0.2820	0.1992
u, b	u, b	0.2362	0.5040	0.4920	0.3980	0.2774
u, b	u, b, c	0.2378	0.5080	0.4880	0.3980	0.2806
u, b	u, b, l	0.2369	0.5080	0.4940	0.4013	0.2758
u, b	u, l, f	0.2315	0.5160	0.4440	0.4007	0.2734
u, b	u, b, l, f	0.2409	0.5160	0.4440	0.3980	0.2768
u, p	–	0.1803	0.4560	0.4040	0.3293	0.2194
u, b	u, pq(1), p	0.1523	0.3640	0.3480	0.2760	0.1912
u, b	u, pq(0.3), p	0.1542	0.3800	0.3600	0.2813	0.1956
u, b	u, pq(0), p	0.1542	0.3920	0.3680	0.2873	0.1968
u, b	u, pq(0.3)	0.2197	0.5000	0.4760	0.3933	0.2660
u, b	u, pq(0)	0.2198	0.5040	0.4780	0.3933	0.2662

u= unigrams included.

b= bigrams included.

s= sensed unigrams used.

b-idf = use idf in Okapi scoring of bigrams.

c= combining 1st pass and 2nd pass scores in a 99 to 1 ratio.

l= length suppression used.

f= fixing top10 first pass rankings.

p= use best passage score as representative.

pq(ξ)= use weighted passages as queries with exponent ξ .

Table 4: Results of experiments on TREC-4.

1st pass	2nd pass	AveP	P5	P10	P30	P100
u	–	0.1466	0.3640	0.3160	0.2340	0.1528
u, b	–	0.1557	0.4120	0.3320	0.2373	0.1560
u, b, s	–	0.1522	0.4120	0.3480	0.2280	0.1518
u, b	u	0.1563	0.3520	0.3100	0.2420	0.1708
u, b	b	0.1139	0.2680	0.2240	0.1733	0.1214
u, b	b-idf	0.1164	0.2960	0.2200	0.1807	0.1266
u, b	u, b	0.1587	0.3840	0.3180	0.2453	0.1718
u, b	u, b, c	0.1643	0.3600	0.3300	0.2593	0.1762
*u, b	u, b, l	0.1585	0.3440	0.3240	0.2600	0.1726
u, b	u, l, f	0.1719	0.4280	0.3320	0.2627	0.1772
u, b	u, b, l, f	0.1685	0.4280	0.3320	0.2573	0.1736

* official TREC-5 result.

Otherwise identical to Table[4].

Table 5: Results of experiments on TREC-5.

TREC-5 was degraded by Knn re-scoring, even though it helps the aveP there. In the future, a far more restrictive selection of terms may be tried. In addition, combining first- and second-pass scores as we did in Section 8 also helps to alleviate this over-expansion problem.

Here is a summary of other results we obtain:

- Morphological analysis produces significant improvements over tokenized words. Sensing produces slight improvement in topK precision, but not aveP.
- Addition of bigrams produces overall improvement in aveP, but surprisingly, it lowers topK precision in TREC-4. Using idf for bigrams produces additional improvement.
- Using best passage scores by themselves degrade performance. Using best passages as expanded queries do not have much effect.
- Length suppression has positive effect on TREC-4, but negative effect on TREC-5.

In the future, we hope to experiment with many of the possible improvements suggested above.

10 Acknowledgements

This work is supported by NIST grant no. 70NANB5H1174. We thank Satya Dharanipragada for assistance in the pre-processing of the corpus, and Robert T. Ward for stimulating discussions and various aspects of computing.

References

- [1] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, M. Gatford 1995. Okapi at TREC-3. In D. K. Harman, editor, *Proceedings of the Third Text REtrieval Conference (TREC-3)*. NIST Special Publication 500-225, 1995.
- [2] C. Buckley, G. Salton, J. Allan, A. Singhal 1995. Automatic query expansion using SMART: TREC 3. In D. K. Harman, editor, *Proceedings of the Third Text REtrieval Conference (TREC-3)*. NIST Special Publication 500-225, 1995.
- [3] B. Merialdo 1990 Tagging text with a probabilistic model. In *Proceedings of the IBM Natural Language ITL*, Paris, France, pp. 161-172.
- [4] E. P. Chan, S. Garcia, S. Roukos, T. Ward. (To be submitted to ACL '97) Bilingual Sensing applied to Information Retrieval
- [5] A. Singhal, G. Salton, M. Mitra, C. Buckley 1995 Document length normalization. Technical Report TR95-1529, Cornell University



The GURU System in TREC-5

Contact

- . Yael Ravin
- . Information Retrieval and Analysis
- . T.J. Watson Research Center, H1-M09
- . POB 704, Yorktown Heights, NY 10598
- . yael@watson.ibm.com (914) 784-7847

System Description

The GURU information retrieval system is an experimental system developed at the T.J. Watson Research Center of IBM to support research in three main areas:

- Storage, indexing, and search of very large collections (hundreds of gigabytes) of full text documents. This includes research into distributed client/server environments and parallel processing.
- Mechanisms for querying the stored documents using free text and for ranking query results.
- Text analysis and information extraction tools for navigation, query preparation and query reformulation. This includes proper name identification, domain specific phrases and abbreviations, acronyms and their full forms (Ravin & Wacholder 1996).

GURU is a client/server system. The server includes an indexer, a search engine, a probabilistic ranking module, a query analysis module and a lexical server. Indexing is fairly standard. Text files are separated into documents, and each document is tokenized into individual words. All words, including stopwords, are indexed with their full position - paragraph, sentence, and word number. We are currently working on indexing proper names as well by integrating a module that identifies occurrences of different variants of the same name in the text and assigns them one canonical form.

At query time, one or more text collections to be searched is specified by the user. If multiple collections are specified, results from the different collections are merged before they are returned to the user. Our probabilistic ranking uses a unique feature called "Lexical Affinity" (LA). LA between two terms is a correlation measure of their common occurrences in text, as defined by Maarek (1991). The occurrences of correlated pairs of words in a document are ranked higher than the occurrences of the individual words over greater distances.

The analyzed query is expressed as an "F" statement. "F" is a formal language which we have developed for specifying different search operations and expansions of the query terms. For example: `label1 { morph (word1) word2 }` determines that the query consists of one query term (corresponding to one label). The curly brackets specify that word1 and word2 are variants of it. Occurrences of either variant will be treated by the search engine as occurrences of the query term. Variants are added manually by the user. The "morph" operator instructs the search engine to expand each word within its scope to all of its morphological forms. Morphological expansion is performed automatically by the engine on all query terms, but it can be over-ridden by the user.

The probabilistic ranking algorithm used by the search engines is based on work reported in Maarek and Smadja (1989) and Maarek (1991).

Participation in TREC-5

In TREC-5 we participated in the adhoc querying task. Our main goal is to optimize the probabilistic ranking algorithm used by GURU. Thus, we submitted four runs:

- Automatic query formulation with one ranking formula (labelled 'd');
- Automatic query formulation with another ranking formula (labelled 'e');
- Manual query formulation with formula 'd';
- Manual query formulation with formula 'e';

Due to various delays, we were unable to perform any training or fine-tuning for TREC. In addition, the person who formulated the queries for GURU was working remotely. As he was not familiar with the system and did not have sufficient time to communicate with us, the queries submitted in August were not properly formulated for the 'd' runs. We are currently re-running with new queries and will report in the final paper on any differences.

The GURU version with which we participated in TREC-5 is very simple: it uses a stopword list of about 250 words and applies automatic rule-based morphological expansion on the query terms. In TREC-5 we did not use proper names, phrases, or any other special data structures or knowledge bases. The system supports cross-collection searching -- the query is evaluated against each collection individually but the statistics are manipulated, to reflect the global statistics of all the collections searched.

References

Maarek, Y. S. "Software library construction from an IR perspective," in *SIGIR Forum*, Fall 1991, 25:2, 8-18.

Maarek Y. and F.A. Smadja "Full text indexing based on lexical relations. An application: software libraries." in N.J. Belkin and C.J. van Rijsbergen, eds, *Proceedings of SIGIR'89*, 198-206, Cambridge, MA, June 1989. ACM Press.

Ravin Y. and N. Wacholder, "Extracting Names from Natural-Language Text", IBM Research Report 20338, 1996.

MercureO2 : adhoc and routing tasks

M. Boughanem	C. Soule-Dupuy
Université de Limoges	IRIT Toulouse
123 AV. Albert Thomas	118 Route de Narbonne
87060 Limoges France	31062 Toulouse France
email : {bougha, soule, trec}@irit.fr tel (33) 61 55 63 22	

1 MercureO2 system

MercureO2 is an object oriented information retrieval system based on connectionist approach. It allows the query formulation, the query evaluation based on propagation of the neuron activation and the query modification based on backpropagation of the user judgements of the document relevance.

1.1 The model

The neural information model used in MercureO2 is shown in figure 1. It represents a set of associated neurons. We distinguish in this model :

- neuron objects
 - an input layer representing the user information needs
 - a term neurons layer
 - a document neurons layer
 - an output layer representing the result of the query evaluation
 - weighted links between the neurons.
- **The MercureO2 neuron object** is characterized by : identifier, input value, output value, output threshold value, methods allowing to compute the input values and output signals, to propagate the signal, to backpropagate the signal,... The functions of this neuron are :
 - Input function : $In(N_i) = \sum_{j=1} Out(N_j) * w_{ji}$
 - Activation (or state) of the neuron at (k+1)stage is :

$$Activation^{k+1}(N_i) = Activation^k(N_i) + g(In(N_i))$$

The initial stage at $k = 0$: $Activation^0(N_i) = 0$

- Output of the neuron : $Out(N_i) = \begin{cases} Activation^k(N_i) & \text{si } Activation^k(N_i) > threshold \\ 0 & \text{otherwise} \end{cases}$

Where N_i is a neuron i and w_{ij} represents the weight of the link between the neurons i and j .

- **The document neuron** represents a particular document descriptor. The document descriptor is a list of terms representing the important concepts in a document. These terms are extracted by an automatic indexing operation.
- **The term neuron** represents an indexing term.

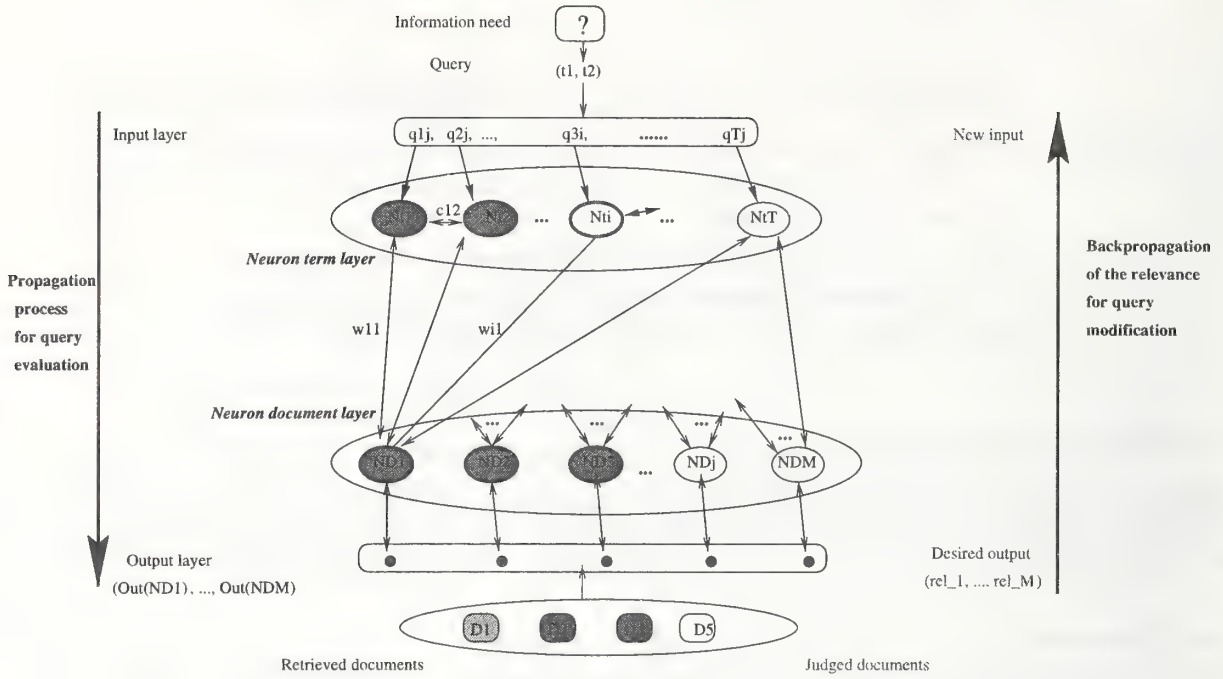


Figure 1: MercureO2 Model

- **The input layer** represents the user information needs. A set of weighted terms representing the query is associated to this information needs. The weight assigned to a term is :

$$q_{ij} = \frac{freq_{ij}}{\max_j(freq_{ij})}$$

The input layer is represented by the vector $Input_j = (q_{1j}, q_{2j}, \dots, q_{Tj})$

- **The output layer** represents the obtained output for the applied Input. It is a vector of output values associated to the activated document neurons. This vector represents the retrieved documents.
- **The links** : The neurons are interconnected by weighted bidirectionnal links. Two kinds of associations are defined : term neuron to term neuron association and term neuron to document neuron association

- **term neuron-document neuron link**

This link represents an indexing link. The weight of this link is a combination of the term frequency (tf) and the inverse document frequency (IDF) [2]. We used the following formula :

$$w_{ij} = \frac{freq_{ij}}{\sqrt{\sum_{k=1}^T (freq_{kj}^2 \cdot \log(\frac{N}{N_k})^2)}} * \log(\frac{M}{m_i})$$

- **term neuron-term neuron link**

This weighted link represents a cooccurrence association [1] . It is defined as follows :

$$c_{ij} = \alpha \cdot \frac{\sum_{k=1}^T (w_{ik} * w_{jk})}{\sqrt{\sum_{k=1}^T w_{ik}^2 + \sum_{k=1}^T w_{jk}^2 - \sum_{k=1}^T (w_{ik} * w_{jk})}}$$

where :

α : a positive constant,

w_{ij} : the weight of the link between the term i and the document j,

$freq_{ij}$: the frequency of the term t_i in the document D_j ,
 M : the number of documents in the collection,
 T : the number of indexing terms,
 m_i : the number of documents containing the term t_i .

1.2 Query evaluation

The query evaluation is based on the neural propagation process. In MercureO2 this evaluation is performed as follows :

1. Transform the user information needs into a list of terms by series of query test processing modules. The result of query processing is the $Input_j = (q_{1j}, q_{2j}, \dots, q_{Tj})$
2. Apply this input to the term layer. Each term neuron computes an input value:

$$In(N_{t_i}) = q_{ij}$$

and then an activation value : $Activation^1(N_{t_i}) = g(In(N_{t_i}))$

An output signal is then generated : $Out(N_{t_i}) = Activation^1(N_{t_i})$

3. These signals are propagated forwards through the network. Each neuron computes an input and an output value.

$$In(N_{D_i}) = \sum_{j=1}^T Out(N_j) * w_{ij}$$

then, $Out(N_{D_i}) = Activation^1(N_{D_i}) = g(In(N_{D_i}))$

The output layer is : $Output_j(Out(N_{D_1}), Out(N_{D_2}), \dots, Out(N_{D_M}))$

These output values computed by the document neurons allow to rank the list of retrieved documents. The output function we used is the sigmoid : $g(x) = \frac{e^x - 1}{e^x + 1}$

1.3 Query modification based on relevance backpropagation

The strategy we use for automatic query modification is based on the neural backpropagation algorithm. The top of retrieved documents are judged by the user. These judged documents will represent the called *Desired output*. To each judged document is assigned a *relevance value*, positive relevance value is assigned to relevant document, a negative value to non relevant document and zero for not judged document. So, the desired output is represented by the vector of the form $DesiredOutput = (rel_1, \dots, rel_i, \dots, rel_M)$ This strategy consists to backpropagate the relevance values from the Output layer to input layer. Thus, The desired output will be applied to the document neurons, activations will be computed by each neurons in the document neurons and then in term neurons. The result of this backpropagation is a new input.

This strategy is defined as follows :

1. Build the desired output : $DesiredOutput = (rel_1, \dots, rel_i, \dots, rel_M)$,
2. Apply this output to the neuron document layer. Each neuron computes an input value : $In(N_{D_i}) = rel_i$
And then an output signal : $Out(N_{D_i}) = Activation^1(N_{D_i}) = g(In(N_{D_i}))$
3. The output signals are backpropagated to term neuron layer. Each neuron term computes an input value : $In(N_{t_i}) = \sum_{j=1}^M (w_{ij} * Out(N_{D_j}))$
and then an output signal $Out(N_{t_i}) = Activation^1(N_{t_i}) = g(In(N_{t_i}))$
4. A new input is then computed according to this formula : $NewInput_j = Input_j + \alpha * Out(N_t)$
5. This new input is applied to the term neuron layer and a new query evaluation is then done. This process (propagation of the input and backpropagation of the desired output) can be repeated several times ; *the iteration number* is a parameter in MercureO2.

Parameters	Routing		Adhoc	
Run Ids	mer-r1	mer-r2	mer-a1	mer-a2
Items used	All	All	All	description
Training Doc	ALL(cat B)	All(catB)		
NbJudgedDocs	50	50	30	30
Coef_Rel (relevant doc)	0.5	0.5	0.5	0.5
Coef_Rel (non relevant)	-0.25	-0.25	-0.25	-0.25
β	0.5	0.5	0.5	0.5
Number of iteration	1	2	1	1
α	0.5	0.5	0.5	0.5
NbAdded terms	60	60	50	50

Table 1: Trec runs parameters

Routing results				Adhoc results			
Run	Best	\geq median	$<$ median	Run	Best	\geq median	$<$ median
mer-r1	14	21	8	mer-a1	10	29	6
mer-r2	27	14	2	mer-a2	7	29	9

Table 2: Comparative TREC Results

For desired output several formulations can be used, for this experimentation we used :

$$rel_i = Coef_Rel * \log_2\left(\frac{NbJudgedDoc}{NbRel} + \beta\right)$$

where :

Coef_Rel : relevance coefficient of the documents (positive for relevant and negative for non relevant document)

NbRel, NbNRel : number of relevant and non relevant documents respectively

NbJudgedDoc : number of judged documents

2 Description of TREC Experiment

Our experiment is done with MercureO2. The initial queries are built automatically and then modified (reweighted and expanded) by using our relevance backpropagation strategy. The terms of the new query are deduced as explained previously. We easily note that after this process the initial query will be expanded and reweighted. For the two experiments (Adhoc and routing) we limited the number of terms to add. We consider the new input and we sort all terms do not appear in the initial query and we add only the *NbAdded* top terms. For TREC 5 experiment we submitted two runs for routing (mer-r1 and mer-r2 with 1 iteration and 2 iterations respectively) and two runs for adhoc (mer-a1 where we used all the items in the topic for query construction and mer-a2 we used only the description item). The values of the different parameters we used are listed in (Table 1).

The results we obtained in comparison with other TREC category B runs are shown in (Table 2).

References

- [1] D., J. HARPER, & C., V. VAN RIJSBERGEN, *An evaluation of feedback in document retrieval using co-occurrence data*. JOURNAL OF DOCUMENTATION, 34(4), 189-216, 1978.
- [2] G. SALTON & M., J. ALLAN, *Automatic Text Decomposition and structuring* RIAO'94 NEW YORK 1994. (USA).

Information Retrieval and Trainable Natural Language Processing

John D. Burger • John S. Aberdeen • David D. Palmer
The MITRE Corporation
202 Burlington Road, Bedford MA 01730
{john, aberdeen, palmer}@mitre.org

Introduction

Existing work on indexing and retrieving documents from large on-line collections has had great success at treating both documents and queries as simple, unstructured collections of individual words (terms) with dependencies among these terms largely ignored. However, natural language text has a great deal of structure. In particular, at a scale close to that of the individual word, there are interactions and dependencies that many IR systems ignore. Those systems that do attempt to capture some of these dependencies do so in rather indirect ways.

MITRE has substantial experience with trainable natural language algorithms, and we believe this powerful approach is complementary to the standard information retrieval paradigm. Developed over a number of years, our technology has been used to good effect in the context of information extraction tasks, such as the Message Understanding Conferences (MUC) [1]. Because our approach is based on automatically training our NLP components on large data sets, resulting in accurate and very fast text processors, and because we believe that empirical evaluation is highly important, we have been interested in classical information retrieval for some time. Our first foray into TREC was begun to explore the possibility of improving IR systems with this technology.

TREC5 system architecture

Our initial goal was to use simple techniques borrowed from our message understanding system to improve a standard IR system (Cornell's Smart [2]), and to evaluate these improvements in the context of the TREC5 NLP track. Unfortunately, the NLP track only involved the ad hoc queries this year, while many of the techniques and technology developed for MUC would seem to be most immediately applicable to the routing task.

This was our first experience with both TREC and Smart. Consequently, we chose Category B participation, and a substantial fraction of our effort was spent in becoming familiar with the TREC tasks, the exigencies of working with that much data (even in Category B), and the workings of Smart. The latter is a complex and sophisticated IR system, the result of several decades of research at Cornell, and is parameterizable in hundreds of ways. Initially, we planned to integrate our own code into Smart, which can theoretically be extended in this way. In the end, however, due to the system's complexity, we decided simply to modify Smart's behavior via pre- and postprocesses. This is far from ideal, of course, due to issues of efficiency and organization, and we plan to pursue a tighter integration with Smart, especially as our additions and modifications become more complex.

The preprocess intended to improve Smart is fairly simple. Its aim is to remove material from the queries that is not useful, and is in fact possibly misleading. Each query is an English topic description, a "statement of need" that includes phrases such as *a relevant document will contain*, and so on. The preprocess is a pattern-matching program, implemented using Flex [3], that strips this kind of material from the queries before they are indexed. This typically improved the average precision of Smart by one point or more on our training set (e.g., from 29.2 to 30.8% in one run).

Another preprocess is used to identify particularly salient keywords and phrases used in each query, based on a number of simple heuristics, beginning with either the title field or the first sentence of the narrative. These keywords are set aside and used in a postprocess that runs after Smart returns an ordered set of documents for each query. The postprocess is employed to improve the ordering of the returned

documents, based on the presence or absence of the keywords and phrases identified as described above.

Neither of the processes makes direct use of our trainable NL techniques yet. However, the means by which we developed the pre- and postprocess are indicative of how we envision using trainable language processors for the next version of our IR system.

In order to develop the preprocessor, which stripped the topics of extraneous verbiage, we first tagged a set of training corpus (TREC topics 151–200) with SGML markup indicating those phrases that we judged to be irrelevant, or even misleading. For example, the following is the description field from topic 151, marked up with SGML for both extraneous material and keywords (discussed below):

<EXTRA>The document will provide information on</EXTRA> <KEY>jail</KEY> and <KEY>prison</KEY> <KEY>over crowding</KEY> and how inmates are forced to cope with those conditions; <EXTRA>or it will reveal</EXTRA> plans to relieve the <KEY>overcrowded</KEY> <KEY>condition</KEY>.

This small corpus constitutes training material, which we used in this case to continually improve a Flex-based tagger, attempting to reproduce the training markup as accurately as possible. The tagger so developed can also be configured to omit material rather than tag it, and thus constitutes the filter on extraneous material that we used in our preprocess. The manual hill-climbing process is less than ideal, and can, in fact, be automated, using our trainable phrase-finding algorithms. We were unable to bring these to bear in time for this year's TREC, however.

The other preprocess described above collects particularly salient keywords and phrases from topics, which are then used post-Smart in an attempt to improve the ranking of retrieved documents. This was also trained in exactly the same way, by attempting to reproduce the <KEY> phrases annotated in the training corpus.

This, then, is how the pre- and post-processes intended to improve Smart were acquired, by manually improving two taggers to reproduce the <EXTRA> and <KEY> annotations in the training corpus. Of course, it is critically important in this process that the person

developing the tagger not over-train on the development corpus, in effect simply memorizing it. The end result in such a scenario would have little hope of performing similarly on new material, e.g., the test topics. The question of whether our automatic algorithms can avoid, or at least minimize, such over-training has been answered in the affirmative for a number of different domains and types of annotation. It remains to be seen whether this is the case for the TREC task.

Taken together, the pre- and postprocess typically improved the average precision of Smart by two to three points of precision. In our development experiments, Smart's baseline average precision was around 29%, which improved to over 32% when our modifications were applied. (These experiments were performed using the "large format" topics, consisting of the title, description and narrative fields of the topics.)

TREC5 results

We have only had time to do a cursory examination of our performance, but it appears that a procedural mistake in applying our simple preprocess degraded Smart's output slightly. Fortunately, the keyword-based postprocess appears to have effected a small improvement over the baseline:

	Average precision	R-precision
Smart	0.1846	0.1782
+preprocess	0.1830	0.1780
+postprocess	0.1896	0.1859
Improvement	0.0066	0.0079

These improvements appear to be positive at all eleven points of recall. Detailed comparisons with the other systems must wait until after the conference, but we are heartened by the fact that our average precision was at or above the median in two-thirds of the queries. Our inexperience with TREC causes us to be wary of making too much of this, however.

Of course, our improvements over Smart's baseline performance are rather minimal, and we are merely happy that they are in fact positive—a (nearly tacit) goal was to avoid degrading Smart's performance!

Future work

Our main goal in participating in TREC5 was simply to gain familiarity with the TREC tasks, and make some small improvements on a baseline system. Having accomplished this, we will now turn our attention to making greater use of the techniques and technology developed for our MUC system. These components were designed to be highly efficient, very robust, and trainable for new types of text and new languages, as well as new domains and application areas [4, 5]. They include part-of-speech taggers and phrase- and name-finders, as well as higher-level language-processing components that can, for example, automatically merge multiple mentions of the same entity (e.g., *John Q. Public*, *Mr. Public*, and *John*). We have used this technology in the context of information extraction tasks quite successfully, fielding one of the top-performing MUC systems.

We believe that our NLP tools will be most effective in the areas of term selection and term normalization. With respect to the former, we will use our high-accuracy part-of-speech tagger to aid in identifying key terms. We also have a variety of trainable name- and phrase-finders (or “phrasers”), as well as a full-fledged NL bracketer/parser (also trainable). We will use these to find appropriate multi-word terms in database documents and queries. We believe that we can find both contiguous and non-contiguous phrases with these methods. By contiguous phrases we mean those formed by immediately adjacent words, for example, nominal compounds and proper names. A non-contiguous phrase is typically composed of a head-modifier pair (or triple), such as a verb and one or more of its syntactic arguments. Our phrasers and parsers have been trained on a large dataset of journalistic prose, and are successful at finding both kinds of phrases with reasonable accuracy. In addition, they are quite fast compared to traditional NLP systems, and so are better suited for large-volume data sets such as the TREC materials.

For term normalization, we will use standard word-stemming techniques, but we are also interested in exploring recent work on word-sense disambiguation. Multi-word terms, whether contiguous or not, will be normalized to head-modifier tuples. As mentioned above, we also have tools for “merging” multiple mentions of the same entity, and we intend to

use these for normalization of personal and organizational names, as well as for other types of phrases. Finally, we have some experience with automatically trained co-reference algorithms [6], and we intend to experiment with these vis à vis term normalization.

We are also interested in using our NL components to aid in resorting the top documents retrieved, in more sophisticated ways than are outlined above. This is similar to what several other systems do, but we posit that specialized language processing may provide a complementary knowledge source. This approach has been successfully used in the speech community, where NL parsers are used to reorder and prune the top N hypotheses of a speech processing component.

References

- [1] John Aberdeen, John Burger, David Day, Lynette Hirschman, Patricia Robinson and Marc Vilain. “MITRE: Description of the Alembic System as Used for MUC-6”, *Proceedings of the Sixth Message Understanding Conference*. 1995.
- [2] Chris Buckley, Amit Singhal, Mandar Mitra, (Gerard Salton). “New retrieval approaches using SMART : TREC 4”, *Proceedings of the Fourth Text Retrieval Conference*. 1995.
- [3] G.T. Nicol. *Flex: The Lexical Scanner Generator*. 1995.
- [4] Marc B. Vilain and David D. Palmer. “Transformation-based bracketing: fast algorithms and experimental results”, *Proceedings of the Workshop on Robust Parsing*. 1996.
- [5] Marc B. Vilain and David S. Day. “Finite-state phrase parsing by rule sequences” in *Proceedings of the 1996 International Conference on Computational Linguistics*. 1996.
- [6] Dennis Connolly, John D. Burger and David S. Day. “A machine learning approach to anaphoric reference” in *Proceedings, Conference on New Methodologies in Language Processing*. 1994.



Using Bayesian Networks as Retrieval Engines

Maria Indrawan^{1*}, Desra Ghazfan², Bala Srinivasan¹

¹Department of Computer Technology, Monash University, Caulfield East 3145, Australia

²Department of Computer Science, Monash University, Clayton 3168, Australia

Abstract

In this paper we discuss Bayesian network implementation for retrieving documents in a text database. We participate in TREC-5 for ad-hoc task in category B. Several problems and possible solutions in implementing large scale text retrieval system using Bayesian network are discussed. The main problems are the existence of loop and large number of parents per-node. The solutions suggested are that of intelligence node and virtual layer. Comparison with other Bayesian approach to text retrieval is also discussed. We show that our approach gives more correct semantic to the retrieval model.

1. Introduction

Uncertainty is present in almost every task that requires intelligent behaviour, such as planning, reasoning, problem solving, decision making and classification. Such uncertainty also exists in many aspects of text retrieval system. Obviously, the employment of uncertainty management techniques that appropriate to the specific needs of text retrieval systems has always been the central issue in the field [Cohen87, Cooper71, Croft80a, Wilson73, and many more]

Text retrieval is concerned with the representation, storage, organisation and accessing information items, which typically include letters, all types of documents, newspaper, facsimile documents, electronic mails, articles, books, medical summaries, research articles, pictures, audio-visual, and so on. Generally, the systems maintain the description of information items rather than the information object themselves. These descriptions usually consist of text describing various attributes of the information items [Salton83], and they intended to give the system the ability to compute the likelihood of the information descriptors in matching the user query.

The process of creating information descriptors does not supported by exact rules, consequently, the task to capture the contents of documents by assigning representations or descriptors can be very difficult. It involves uncertainty. Similarly, formulating a query from user's internal information need involves uncertainty. One of the solutions to this uncertainty problem is by using probability theory in estimating the likelihood of matching between the document descriptors and user's formulated query. Among the probability methods for handling uncertainty, Bayesian network is the most dominant method in the AI community and is considered the most common representation scheme for probabilistic knowledge.

Bayesian network approach is not new to text retrieval field. Among other implementations of Bayesian network text retrieval systems, the work of Croft [Croft80b], especially the ones that were carried out in collaboration with Turtle [Croft80b, Turtle90, and Turtle91] are the most representative. Other works can be categorised either as refinements to Turtle and Croft's work, as a logical extension from the field of hypertext or as having a bit of both [Lucarel93]. Our method which will be explained in section 2 of the paper provides different implementation of Bayesian network for text retrieval. Our implementation provides a correct semantic meaning for the inference process involved in text retrieval system. Section 3 provides details comparison between our methods and the existing model. The implementation of Bayesian network for TREC experiments is discussed in section 4.

2. Bayesian Network for Text Retrieval

The life-cycle of a text retrieval system begins with the development of document representations. In a Bayesian network-based text retrieval system, the process is characterised by the construction of the document network. It represents a static knowledge about the document collection

* Corresponding author. E-mail: maria@broncho.ct.monash.edu.au

and only need to be created once. In real-world applications, the document network can be modified to characterise new state of document collection as documents are added or omitted from the collection. Another network needs to be built but unlike the documents network, this network is dynamic because it represent user's information need. This dynamic network is called the query network.

The document network consists two layers of nodes at the minimum. The first, top most layer makes up of a universe of keywords. In text retrieval system this layer represents the collection's dictionary because it contains every keyword known to the collection. The next layer down comprises of concepts, events, or any sensible entities that can be generated by the combination of any arbitrary keywords. One of possible entities is a document. The associations between the entity documents and the keywords are characterised by the existing of direct link from the keywords to the documents, thus it is easy to see that the keyword nodes cause the document nodes to exist. For example, in Figure 1 the keywords "A", "B", and "C" make up document "1", while the combination of "A", "C", and "D" forms document "2".

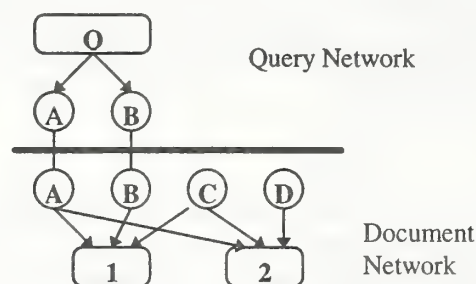


Figure 1. Example of Bayesian network for text retrieval

The query network explains user's information need to the system. The query network comprises a single rooted network that symbolised the user's information need. This information need is explained further by a set of keywords which forms the description of the query. This relationship is signified by the existence of directed links emanating from the information need node to the description node. Figure 1 shows a user query represented by the node Q which can be described by the keywords A and B . The query network is dynamic, and it is created whenever a user query the collection. The query network is dynamic because it is a temporal network. It only exists during the querying of the document collection. Once the result has been obtained, the query can be discarded or modified to represent another information need.

The matching process is performed by propagating belief down from the evidence arrives at the systems that is of a query node to the document nodes. The final belief values at the document layers reveals the documents' probability of relevance with respect to the query submitted. A sequence of ranked documents in decreasing order of relevance to the query is then returned to the user as the possible answers to the user's query.

3. Comparison with an Existing Implementation.

Contrary to our approach described in section 2, Croft and Turtle's has different basic causal topology and underlying assumptions of a evidence. In their model, they assumed that the documents triggered the existence of the keywords and the queries are constructed by the keywords. Using this assumption, the values of $P(\text{Relevant}|\text{document}_i, \text{query}_j)$ are obtained at the query node. The causal

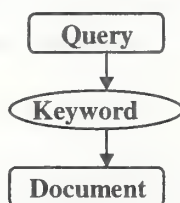


Figure 2.a.

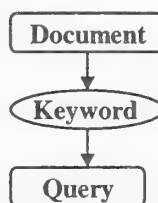


Figure 2.b

Figure 2. Causal topology contrast

topology contrast can be viewed in Figure 2. The network on Figure 2.a is our implementation approach while Figure 2.b is Croft and Turtle's implementation. In the following paragraph we will contrast the two approaches.

Bayesian network can be described as a system containing two types of entities: *evidence or cause* and *hypothesis or effect*, and their relationship can be summarised by the following formula:

$$P(E | C) = \frac{P(E, C)}{P(C)} = \frac{P(C | E)P(E)}{P(C)} \quad (1)$$

In the graph of Bayesian network, the direction of an arc is placed from the *Cause (C)* to the *Effect (E)*. In practice, the direction is placed from the evidence, an event which we can gather either directly or from the use of other rules to the desired conclusion. It is also common practise to determine the directionality of an arc according to the which conditional independencies are easiest to identify or according to which conditional probabilities are easiest to ascertain [Neap89]. The definition and determination of "real" causes in a text retrieval can lead to different model of Bayesian network. Applying $P(\text{Relevant} | \text{document}_i, \text{query}_j)$ to the Bayesian network implementation in Figure 2., we obtain relationship as depicted in Figure 3.

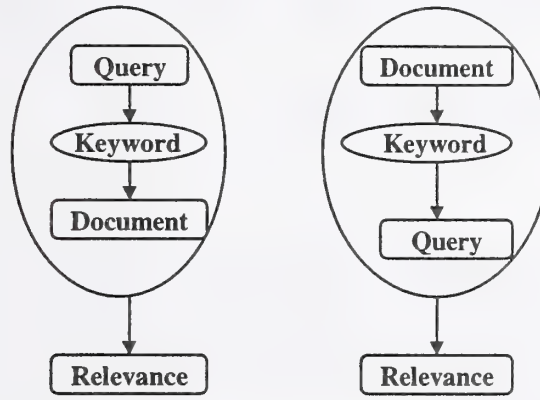


Figure. 3.a

Figure.3.b

Figure 3. The relevance graph.

The relevance node represents an event whereby the document is judged relevant to a query. The graph on in Figure 3.a can be translated into:

$$\begin{aligned} P(\text{Relevant} | \text{document}_i, \text{query}_j) &= P(\text{Relevant} | \text{document}_i, \text{keyword}, \text{query}_j) \\ &= P(\text{document}_i, \text{keyword}, \text{query}_j) \end{aligned}$$

And by only conditioning the probability values on the evidence set (query), the above equation can be simplified further into:

$$P(\text{Relevant} | \text{document}_i, \text{query}_j) = P(\text{document}_i | \text{query}_j) \quad (2)$$

Using the same procedure, the graph in Figure.3.b simplifies to:

$$P(\text{Relevant} | \text{document}_i, \text{query}_j) = P(\text{query}_j | \text{document}_i) \quad (3)$$

The belief values forwarded to the relevance nodes are the results of belief propagation process triggered by the arrival of the new evidence. The arrival of a new evidence for Figure.3.a (our approach) is indicated by the introduction of a query into the system. The model depicted by figure 3.b obtains the evidence by instantiating one document in the collection. In our approach, the relevant value to be measured is the one that lying on the document nodes. Our belief in the documents would meet the query is the hypotheses that we have to verify. Thus, the ranking of the documents is obtained at the document's nodes. It is worth noting that our method requires only one step to propagate the belief triggered by a new query to all documents.

The approach adapted in Figure.3.b measures the relevance on the query node. The hypothesis to be verified lies on our belief in the query meets a document's explanations. Thus, in this model, a set of relevant documents is obtained by ranking the belief value on the query node in decreasing order. The fact that there is only one query node to process and there are lots of documents to evaluate, the evaluation process is repeated many times for the whole process of inference of a query.

Applying the equation (1) to both of equation (2) and (3) reveals another intriguing findings:

$$P(\text{document}_i | \text{query}_j) = \frac{P(\text{query}_j | \text{document}_i)P(\text{document}_i)}{P(\text{query}_j)} \quad (4)$$

$$P(\text{query}_j | \text{document}_i) = \frac{P(\text{document}_i | \text{query}_j)P(\text{query}_j)}{P(\text{document}_i)} \quad (5)$$

The denominators in equation (4) and (5) are the normalisation factor of the equation. In the process of answering an arbitrary query, equation (4) uses the same normalisation factor on every document matching process for that query. Equation (5) on the other hand, gives different normalisation factor for each observed document in the same query. Recall that in text retrieval system, the collection is fully evaluated for each query introduced to the system. Hence, a common denominator is preferable while processing a query. And since there are more than one documents to evaluate, document belief value cannot be used as the common denominator. The query belief value is the only common values shared by the processes. This demonstrates that an implementation exhibits the feature of equation (4) will give a correct result.

The effect of applying different denominator can be seen in the following example: suppose there are three documents in the system: the book A, the thesis B and article C. The mapping of document contents on the keyword space and query relevance towards documents are given in Figure 4. A quick visual observation on the graph reveals these facts : Book A has around 50% of Q, thesis B has around 30% and article C has very little of Q. Note that query Q has around 30% of A, around 70% of B and, most of C. Intuitively, we will choose A, followed by B and maybe C as the list of relevant documents in decreasing order of its relevance to the query. However, if we applied equation (5) yields the following order: C,B and A.

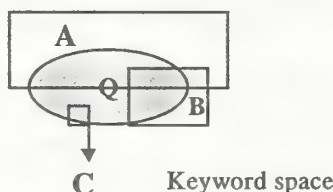


Figure 4. Query and document mapping in keyword space

4. Experiment and Results

We spent most of our effort in this year TREC to “scale up” our system since it is the first year we have joined the conference. One automatic run for Ad-Hoc task was submitted. We use SMART¹ text retrieval System version 11.0 to index the documents.

4.1. Index creation.

The index files were created from Wall Street Journal text distribution from NIST via the following process. First, the text is preparsed from the SGML-format into internal representation of

¹ The SMART.11.0 was made available from Cornell University.

update their belief value because the belief in d_l was changed due to the message comes from them, namely π_{k_1} and π_{k_2} . Only node k_3 will use the λ_{d_l} to update its belief value. By stopping the flow message back in k_1 and k_2 , we practically break the loop in the network.

We have shown how we handle the fully connected network that we use to implement the bayesian netowrk for text retrieval using the intelligence nodes. However, we still have to overcome another problem faced by the implementation model, that is of large number of parents pernode. To calculate the probability of a given nodes x which has n number of parents, we need to have a matrix of the size of 2^n . In text retrieval, one document may have more than 200 keywords, thus we may need to

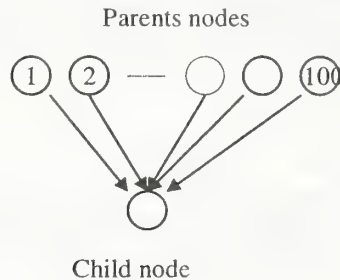


Figure 6.a. Original network

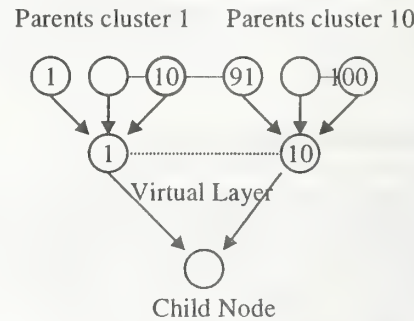


Figure 6.b. Modified network with virtual layer

provide 2^{200} matrix. It is a very costly calculation if not impossible to implement. To overcome this problem, we introduce a virtual layer into our model. The parents of a particular node are distributed into number of groups. The groups then link to a virtual node. The virtual nodes act as the summarised node for the corresponding cluster and links to the document nodes. Figure 3.a shows the network of a node with 100 parents. It requires a matrix of the 2^{100} . With virtual layer the size of the matrix is reduced to 11×2^{10} as in Figure 3.b.

Introducing the virtual layer reduces the computation to a certain extent, however we find that the document network is still very large, and due to the limited resource that we have, we have to implement further "filtering" to the document network. We assign "false" beliefs to all the keywords not involved in the query, thus we expected that the level of recall will be reduced drastically. For example, consider the network in figure 5, node k_3 will be assigned to false or $P(k_3)=0$ because the query Q does not have direct link to node k_3 . If there are other documents apart from d_l contain keyword k_3 , it will not get retrieved.

	\geq Median	$<$ Median
Number of relevant retrieved	24	18
Average precision	16	26

Table 1. Experiments result for ad-hoc task
Category-B using short topic.

The results published TREC organiser confirmed our expectation about the recall and precision level. By taking out all the keywords not involved in the query we have omitted the possibility of retrieving the documents that do not contain the keywords but may share the same concept with the query. This is not the ideal situation. In the preliminary experiments using smaller collection such as ADI, MED and CACM, we managed to include all the keywords and documents in the calculation process. The result has been very promising[Ghazfan96]. Further optimisation of the network structure need to be investigated to accommodate very large text collection as in TREC collection.

5. Conclusion

In this paper we have presented retrieval model for text retrieval system based on Bayesian network. Because of probabilistic nature of the inference process in text retrieval, Bayesian network is suitable for the system. The TREC-5 experiments may not show a very good result. However, this mainly influenced by omitting the keywords in the collection to reduce the size of the network due to our limitation. Preliminary experiments in traditional collection CACM, ADI and MED have shown promising performance of the system. To implement a bayesian network for a large text retrieval system, optimisation of the network structure need to be considered.

References

- [Buckley85] Buckley,C., "Implementation of the SMART Information Retrieval System," Dept of Computer Science, Cornell University, no TR-85-686, 1985
- [Cohen87] Cohen, P.R and Kjeldsen, R., "Information Retrieval by Constrained Spreading Activation in Semantic Network," *Information Processing Management*, Vol. 23, no. 2, 1987, pp. 225-268.
- [Cooper71] Cooper, W.S., "A definition Relevance for Information Retrieval," *Information Storage and Retrieval*, Vol. 7, 1971, pp. 19-37.
- [Cooper90] Cooper,G.F., "The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks", *Artificial Intelligence*, Vol. 42, 1990, pp. 393-405
- [Croft89] Croft,W.B., and Turtle, H., "A Retrieval Model Incorporating Hypertext Links," In *Proceeding of the Hypertext'89 Proceedings*, 1989, pp. 213-224
- [Ghazfan96] Ghazfan,D., Indrawan,M.,Srinivasan B, " *Towards Meaningful Bayesian Network for Information Retrieval*", In *Proceeding of IPMU*, 1996, pp.841-846
- [Lucarrel93] Lucarella, D. and Zanzi, A., "Information Retrieval from Hypertext: An Approach Using Plausible Inference," *Information Processing and Management*, vol.29, no.3, 1993, pp. 229-312
- [Neapolit89] Neapolitan, R.E., *Probabilistic Reasoning in Expert Systems:Theory and Algorithms*, John Wiley & Sons, Inc., 1989
- [Pearl88] Pearl, J., *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann.,1988
- [Salton83] Salton,G. and McGill, M.J., *Introduction to Modern Information Retrieval*, McGraw-Hill Inc., 1983
- [Turtle90] Turtle, H.R., *Inference Network for Document Retrieval*, Ph.D. dissertation, Computer and Information Science Dept., University of Massachusetts at Amherst, Mass., 1990
- [Turtle91] Turtle,H.R. and Croft, W.B., "Evaluation of an Inference Network-Based Retrieval Model," *ACM Transaction on Information Systems*, Vol. 9, no. 3, July 1991, pp.187-222
- [Wilson73] Wilson,P., "Situational Relevance, " *Information Retrieval and Storage*, Vol. 9, 1973, pp. 457-471



CRL English Routing System for TREC-5

Jim Cowie and Zhiqiang Guan

Computing Research Laboratory

New Mexico State University

Las Cruces, NM 88003

jcowie@crl.nmsu.edu, zguan@crl.nmsu.edu

Overview

The routing system described here was initially developed as a final filter to rank documents being produced by multiple queries given to a Boolean retrieval system. This system was still being developed by the time the TREC-5 deadline arrived and it was decided to use the filtering component on its own for the routing task. The router was trained using the relevance judgements from previous evaluations. It is based on a theoretical approach for discriminating between two different types of documents developed by Guthrie and Walker [1]. The generalizations made to extend this approach to multiple document types are not, however, particularly well founded. The system is a true router, each document is handled individually and matched against a profile for each topic to decide if it should be rejected or included. The results for the method are poor. The most likely cause for this is that any word in a profile can count multiple times towards success or failure. The method is similar in many ways to a vector based retrieval system, with a vector for each topic being matched against a word vector for the document.

Method

Word and two word frequency lists are produced for each relevant document set and for "general" text (all documents). These are sorted in descending frequency and words (or bi-grams) chosen which occur in the first n words in the list for relevant documents and after the first m words in the list for non-relevant documents. These two values were finally set at 500 and 1000. A list of words is chosen in the same way to represent non-relevant doc-

uments. Thus we have a list of words occurring frequently in relevant documents and infrequently in relevant documents and a second list of words occurring frequently in non-relevant documents and infrequently in relevant documents. We know the frequencies of each word in each list for both types of documents and the total number of words in both types. We can thus calculate the frequency of occurrence of each type of word; relevant, non-relevant, and words which are not considered significant, for both document types (relevant and non-relevant).

These are used to estimate if a documents word profile belongs to relevant, or non-relevant by calculating

$$(n1 \log p11 + n2 \log p12 + n3 \log p13)$$

to estimate if a document is relevant, where $n1$ is the number of "relevant" words and $p11$ their relative frequency in the relevant document training data, $n2$ is the number of "non-relevant" words and $p12$ their relative frequency in the relevant document training data and $n3$ is the number of other words and $p13$ is their relative frequency in the relevant document training data. A similar measure is made to estimate if the document is non-relevant. If the difference between the two measures is greater than zero the document is passed to a second ranking filter based on bi-grams. This second does not reject documents, but gives each a score by which they are ordered. This order was based on various trials which seemed to indicate that a word based measure gave better recall, and the bi-gram based method gave better precision.

Word Lists

The word lists generated are about 100 items long and many of the words are obviously related to the topic. Temporary occurrences (e.g. a particular incident occurring in a certain location) also show up in the lists and some method is needed to remove these, as they are in fact topic independent. Alternatively each word should only be allowed to count once in a document and the relative frequencies used should be number of documents containing a word divided by the number of documents. We intend to experiment further with this technique.

Example Lists

Partial lists are given below for the poaching (77) and smoking cessation (173) topics.

Topic - 77 - Top relevant words

573 IVORY
426 WILDLIFE,
254 POACHERS,
253 FISHING
245 ELEPHANT
237 POACHING
232 ELEPHANTS
222 FISH
199 SPECIES
199 HUNTING
196 ANIMALS
184 AFRICAN
175 ENDANGERED
166 BAN
164 ILLEGAL
146 AFRICA
144 BEAR
126 PARK
124 KENYA
123 POPULATION
115 RHINO
112 COAST
111 TUSKS
105 ANIMAL
99 CONSERVATION
97 EST

Topic - 77 - Top relevant bi-grams

254 U_S
100 ENDANGERED_SPECIES
91 NR_EST
74 NR_EDT
67 EST_R
67 EDT_R
56 NATIONAL_PARK
54 FISH_WILDLIFE
49 AFRICAN_ELEPHANT
48 WILDLIFE_SERVICE
45 IVORY_TRADE
43 ELEPHANT_POPULATION
39 TRADE_ENDANGERED
38 NORTH_KOREA
37 NORTH_KOREAN
35 NATIONAL_PARKS
33 WORLD_WILDLIFE
32 WILDLIFE_FUND
29 S_FISH
29 ELEPHANT_IVORY
28 GALL_BLADDERS
27 IVORY_IMPORTS

Topic - 173- Top relevant words

2779 SMOKING
1624 TOBACCO
1115 BAN
635 SMOKE
635 ADVERTISING
624 FLIGHTS
613 CIGARETTE
568 SMOKERS
347 AIRLINE
341 ADS
331 CIGARETTES
309 AD
289 AIRLINES
256 FLIGHT
222 DOMESTIC
210 PHILIP
184 RESTRICTIONS
181 MORRIS
180 RJR
179 TRANSPORTATION
172 CANCER
165 MEASURE
160 PASSENGERS
155 SPENDING
155 LEGISLATION
150 NORTHWEST
149 INSTITUTE
147 BANNED
146 SAATCHI

Topic - 173 - Top relevant bi-grams

1461 U_S
336 SMOKING_BAN
262 ANTI_SMOKING
174 TOBACCO_PRODUCTS
173 PHILIP_MORRIS
170 BAN_SMOKING
166 TWO_HOURS
157 TOBACCO_INDUSTRY
120 HOURS_LESS
113 TOBACCO_ADVERTISING
110 DOMESTIC_FLIGHTS
107 NR_EDT
102 TOBACCO_COMPANIES
96 NR_EST
95 PRODUCTS_TOB
95 NON_SMOKERS
90 AIRLINE_FLIGHTS
88 SURGEON_GENERAL
85 SMOKE_FREE
80 SMOKING_DOMESTIC
79 NON_SMOKING
78 SMOKING_AREAS
75 FLIGHTS_TWO

Performance

The combined recall-precision measure over all topics was 0.20. This is a poor level of performance and indicates serious flaws in the method.

Acknowledgement

The work described here was funded by the DoD contract MDA904-91-C-6153.

Reference

[1] Guthrie, L, Walker, W., and Guthrie J., Document Classification by Machine, MCC-92-235, Computing Research Laboratory, New Mexico State University, 1992.

NEW EXPERIMENTS IN CROSS-LANGUAGE TEXT RETRIEVAL AT NMSU'S COMPUTING RESEARCH LAB

*Mark Davis
Computing Research Lab
New Mexico State University
Box 30001/3CRL
Las Cruces, NM 88003
madavis@crl.nmsu.edu*

ABSTRACT

In Cross-Language Text Retrieval, queries in one language retrieve documents in other languages. Query translation is the least expensive approach to the retrieval task when compared to full document translation. The simple combinatorial properties of vector-based text retrieval systems simplify the translation task enormously, reducing most translation to the correct substitution of equivalents from a bilingual lexicon or corpus. New experiments are presented on methods for selecting among potential equivalents from a bilingual lexicon, including one fully-automatic method that achieves 73.5% of the performance of a monolingual system operating on the same retrieval task.

Introduction

In Cross-Language Text Retrieval¹ (CLTR), queries in one language retrieve documents in other languages that are related to that query. Between translating all of a document corpus into one single language prior to indexing or, alternatively, simply translating the user query at query time, translating just the query is generally considered the least expensive in terms of resources, and potentially more accurate given the current state of machine translation technology.

Early experiments by Salton (1971) demonstrated that CLTR could do as well as monolingual approaches given certain experimental constraints. Primarily, this meant preparing a transfer dictionary in advance that contained precise translations of terms in the query language. Homographs and polysemous terms were not a significant obstacle because the terminology in the dictionary was disambiguated by a human in advance of the retrieval experiment. The added problems introduced by translation pragmatics similarly dissolved.

In recent years, however, it has become apparent that the issues in practical, fully-automatic CLTR systems are substantially more complex than originally

conceived. Foremost among these issues is the question of whether the linearity of vector-based retrieval systems leads directly to the application of term-for-term translations. This issue is already being answered by the realization that phrases are not always reducible in machine translation or CLTR systems (Hull and Grefenstette, 1996). A related issue is whether the information retrieval model makes corpus-based term disambiguation practical. Thus far, only mixed results have been achieved for large-scale evaluations of CLTR systems, although TREC multilingual corpora have made further studies much easier (Davis and Dunning, 1995; Hull and Grefenstette, 1996). In recent years, NLP tools like parts-of-speech taggers have improved to beyond the 90% performance level. For CLTR systems, this means that these tools are no longer an unknown quantity, and any performance gains due to using them should become less ambiguous.

Parallel corpora have been shown to be useful for disambiguating monolingual term senses in limited tests (Leacock, Towell and Voorhees, 1993). Parallel corpora have also been used for training statistical text models for translation (Church and Mercer, 1993), and parallel corpora have been implicitly applied to the CLTR disambiguation problem by Landaauer and Littman (1991) who generated query translation matrixes using Latent Semantic Indexing. Davis and Dunning (1994, 1995)

¹ At SIGIR 96, participants in the Cross-Linguistic Information Retrieval Workshop voted to refer to text retrieval with differing query and document languages as Cross-Language Text Retrieval to reduce some of the confusion that other names have caused in the past. I follow this convention in this paper.

applied evolutionary programming methods to attempt to refine Spanish translation of English queries by iteratively comparing the retrieval profiles of English and Spanish queries over a parallel corpus. In Davis and Dunning (1994), a transfer dictionary was used to create the Spanish queries, but no large-scale retrievals were performed, and the later work (Davis and Dunning, 1995) used initial Spanish equivalents derived directly from a parallel corpus. Results from the latter were shown to be comparatively poorer than even the full transfer dictionary methods. In both cases, the evolutionary optimization methods were computationally expensive, requiring around 50,000 retrievals per query to achieve acceptable levels of optimization.

An alternative model of CLTR using parallel corpora is to attempt to disambiguate the Spanish equivalents by comparing their retrieval results one at a time against the English query retrieval results as a whole. Vector-based retrieval models use linear combinations of term occurrence features. As a result, the subspace of the Spanish term-document space projected along an axis of a given equivalent may be adequate for determining the correct equivalent for an English term.

A yet further alternative is to make use of NLP tools like parts-of-speech (POS) taggers. Wilks (1996) has suggested that POS tagging may be combined with full lexicons to disambiguate up to 95% of English. The role of a tagger in a CLTR system then becomes pre-selecting equivalent sets for each query term based on the POS of the tagged query terms. Corpus-based methods can then be applied to the remaining equivalent set where ambiguities still exist to further refine the translation.

In this paper, results are presented for the disambiguation of transfer dictionary equivalents using parallel corpora, equivalent sets selected by matching parts-of-speech (POS) generated by an automatic POS tagger and a bilingual lexicon, and combined results from the two methods. Disambiguation appears to be effective even without the added complexity of considering the entire range of possible equivalent combinations as was done in the evolutionary programming models. The effectiveness of corpus-based disambiguation alone appears mixed in these experiments. The best performance is to be found in disambiguating POS-tagged queries over a parallel corpus, achieving 73.5% of the performance of the original monolingual queries on the same retrieval task. The addition of corpus-based disambiguation represents 6% of this figure, despite the fact that the parallel corpus was drawn from distinct domains of documents.

The experiments reported in this paper are all based on making use of English, human-produced translations of TREC Spanish topic descriptions. The CLTR system

translates these query descriptions into Spanish in a fully-automatic manner and the new Spanish queries are run against the TREC Spanish document corpus. The best we could reasonably hope for would be that the automatic query translations performed at least as well as the original Spanish queries over the same document set. The performance of the monolingual Spanish topics then serve as a comparative baseline for automatic translation methods. In an operational setting, a CLTR system user would be creating, for example, English queries to retrieve documents in multiple languages. The translation process would convert the query into the range of languages that are represented by the document corpora of interest to the user, and the retrieved documents could then be submitted to a translation staff or to a machine translation system for a quick gloss of the document contents.

Recuerdo: A Spanish Retrieval Engine

In order to perform our CLTR experiments, we needed a retrieval engine with competitive performance characteristics. The system also needed to be able to operate over parallel corpora for disambiguation in addition to working on a monolingual document collection. Current Spanish monolingual retrieval systems are primarily vector-based (using variants of tf-idf document and term weighting), inference-net based, and derived from logistic regression of a retrieved document set. The flexibility of the vector-based tf-idf approach suggested that it was a reasonable approach. Further, a vector model is an inherently linear combination of term weightings, making the substitutions of term equivalents in a CLTR scenario straightforward, with special handling of phrasal components an added option that can be accommodated easily without significant modification of the system.

The *Recuerdo* system developed at CRL has some substantial modifications over the Smart system from Cornell. Among these was the development of new Spanish stemmer based on the Porter stemmer model that contains 145 rules for stemming Spanish terminology. The complexity of irregular Spanish verbs was partially handled within this framework, although it was decided to do without specifying irregular verb paradigms precisely to maintain the speed of the stemming algorithm. The effectiveness of this approach has only been tested within the framework of the retrieval experiments presented in this paper.

The system is capable of indexing at around 200 Mb per hour, Spanish or English, and creates indexes of around 0.5 the size of the original document collection. Posting vectors are incrementally written to B-tree databases to conserve memory and then merged at the end of the process without the necessity of sorting the individ-

ual posting sets. Additional options allow for the creation of a database of compressed document signatures which are useful for experimenting with automatic document feedback, although these features were not applied in the results presented in this paper.

For CLTR applications, the system can read multiple indexes for parallel texts, and can perform comparisons between retrieval results for queries across parallel corpora using either a transfer dictionary or the direct extraction of equivalents from the parallel corpus. The system can also perform term expansions by finding the subset of terms it has encountered at index time that have up to a certain number of character differences with the source term. This fuzzy matching capability is used for finding cognates of query terms in CLTR where no dictionary or parallel corpus term is available.

For the system to operate in a fully CLTR mode, it is necessary to supply kill lists in both query source and target languages, transfer dictionaries, an indexed target collection and indexed parallel text collections. For these English-Spanish CLTR experiments, the Collins bilingual dictionary was used as a transfer dictionary and one year of the UN parallel corpus was used as a parallel text collection. POS-tagged queries were handled by a second version of Collins that included POS groupings of lexical items.

Collins Bilingual English-Spanish Dictionary

Collins is a comprehensive bilingual dictionary containing around 50,000 English headwords. For this experiment, English headwords and a subset of the collected *equivalents* and sense discriminating terminology were extracted. Equivalents from homographs and discriminating terms were conflated after case normalization and Porter English headword stemming. Duplicate equivalents were not removed from the conflated term set. The Spanish equivalents were case normalized and stemmed using a Spanish variant of the Porter stemming algorithm developed at CRL. For this experiment, phrasal headword entries were also discarded.

After this preprocessing, 23,932 English headwords remained with an average of 1.394 equivalents per headword (variance of 0.648), with the largest headword having 16 equivalents. This set was checked by a Spanish-fluent graduate student against the original Collins entries, who added missed equivalents to English headwords that also appeared in the queries. The student was provided only the pooled terms from the 25 TREC Infotel queries and was instructed to make certain that the equivalent sets were complete.

A second transfer dictionary was prepared for use in the POS-tagging experiments. The Collins markup

for nouns, transitive and intransitive verbs and adjectives were mapped onto the Penn Treebank POS tagset by conflating all Collins nouns to NN tags, all Collins verbs to VB tags and all Collins adjectives to JJ tags. Headwords with multiple parts-of-speech became separate lexical entries in the resulting dictionary, with the headword prefixed by the tag and an underscore.

The UN Parallel Corpus

The 1991 UN parallel documents were automatically aligned (Davis, Dunning and Ogden, 1995) resulting in 97,594 alignment pairs at the sentence or double-sentence level. The English documents contained 91,915 unique terms out of a total of 4,483,677. On the Spanish side, there were 122,827 unique terms in a total of 5,259,124. The alignment process has previously been estimated to be 83% correct, although a comprehensive evaluation of the UN alignments was not performed.

The 1991 UN document set was chosen because it was suspected that current issues might be better represented by the most current document set from the UN collection which includes years 1988 through 1991.

The English set of aligned texts was indexed using *Recuerdo* with the Porter stemmer variant and case normalization. The Spanish set was similarly indexed simultaneously, with alignment blocks sharing document numbers between the parallel sets. The resulting indexes occupied a total of 77 Mb of disk space, including inverse term token-term dictionaries for testing purposes. The indexing took approximately 20 minutes on a Sparc 5.

The MITRE Parts-of-Speech Tagger

For the POS-based experiments, the MITRE English POS tagger was applied to TREC-5 queries. The English description field of the SGML markup was modified slightly by adding <s> and </s> tags at the beginning and end of the field. For example, Trec-5 Spanish query 56 is:

```
<top>
<num> 56 </num>
<title> Swine Fever
<E-desc>
How has the threat of swine fever affected
international trade?
</E-desc>
<E-narr> Narrative:
A relevant document will contain information
detailing some effect caused by
fears of swine fever.
<S-desc>
¿Qué efecto ha tenido en el comercio inter-
nacional la enfermedad "fiebre
porcina?"
```



```

</S-desc>
<S-narr> Narrative:
Un documento relevante tendrá información
sobre el efecto que ha tenido el
temor a la fiebre porcino en el comercio
internacional.
</top>

```

The tagged version of the query is (just the <E-desc> field is shown for brevity):

```

<top>
<num> 56 </num>
<title> Swine Fever
<E-desc><s>
<lex pos=WRB>How</lex> <lex pos=VBZ>has
</lex> <lex pos=DT>the</lex> <lex
pos=NN>threat</lex> <lex pos=IN>of</lex>
<lex pos=NNS>swine</lex> <lex pos=NN>fever
</lex> <lex pos=VBD>affected</lex> <lex
pos=JJ>international</lex> <lex
pos=NN>trade?</lex>
</s></E-desc>
...
</top>

```

A further filtering step was performed by collapsing the spectrum of noun and verb tags generated by the MITRE POS tagger to JJ, NNP, NN, VB, CD and FW, and then prefixing each query term with the POS tag. Other tags were eliminated. For the query above, this resulted in:

```

<top>
<num> 56 </num>
<title> Swine Fever
<E-desc><s>
VB_has NN_threat NN_swine NN_fever
VB_affected JJ_international NN_trade
</s></E-desc>
...
</top>

```

Overall, the performance of the MITRE tagger on the English description fields was very good. Among the tagset that remained, there were 8 errors by the MITRE tagger over a total of 222 labelled terms in 25 queries, resulting in a 3.6% error rate on query tagging. Notable errors included: "in" was identified as a Foreign Word (FW), "steps" was incorrectly identified as a Verb (VB) twice, and "extinction" was identified as a Verb (VB) once.

Disambiguation and Retrieval

Figure 1 diagrams the method used to retrieve the baseline retrieval results. The Spanish monolingual query retrieves Spanish TREC documents in this approach.

Figure 2 shows the simple extension to this procedure that involves simply replacing each English query

term with all of its Spanish equivalent terms from the Collins bilingual dictionary.

Disambiguation of term equivalents was performed by selecting the best Spanish equivalent or equivalent set for each English query term. In the corpus-based disambiguation experiments, the criterion for equivalent selection was based on examining the distribution of English query terms and candidate Spanish equivalents across aligned parallel texts. For POS-based disambiguation, the approach was to select sets of equivalents from a bilingual lexicon that match the POS of the English query term.

The corpus-based system scored the inner products of weighted document vectors for the English and Spanish retrievals over parallel documents, selecting the term with the highest score. This process thus favored Spanish equivalents that had the most in common with the English query results. This process is diagrammed in Figure 3.

If the English term had no dictionary entry, a fuzzy match was done between the English term and the target retrieval term database to discover potential cognates in the target index. The fuzzy matching process first used an edit distance of zero and, if the term was not found, used an edit distance of two characters.

Adding the fuzzy matching process addressed two fundamental problems associated with general transfer dictionaries: limited coverage and dated material. Specialized terminology is often of neo-latin origin or loan words. In many cases, proper nouns do not have a translation but become loan words in the translation process. The fuzzy matching process makes matching these terms function automatically. If the term is not in the dictionary, an equivalent can often be directly resolved from the target document collection.

The POS disambiguation approach used the MITRE tagger markup of the English query to select among candidate equivalents in the Collins dictionary. This process is diagrammed in Figure 4. NNP tags were handled as special cases of nouns. If equivalents were found in the dictionary under the NN tag category, corresponding to all Collins nouns, then the substitution was performed. Hence, NATO was correctly translated in Collins as OTAN. If no equivalent existed, however, as was the case for proper names, NNPs became their own equivalents. Hence, MERCOSUR translated as itself.

The combined approach is diagrammed in Figure 5. The resulting equivalent sets for each English query term after POS disambiguation were submitted to the corpus-based disambiguation engine in this approach. For the AFP TREC-5 query set, there were 166 unique query terms and 428 equivalents, for an average of 2.58 equivalents per English term. The corpus-based method

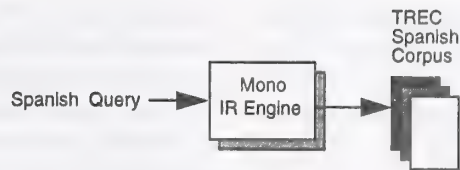


Figure 1: Monolingual Spanish Retrieval for comparison baseline.

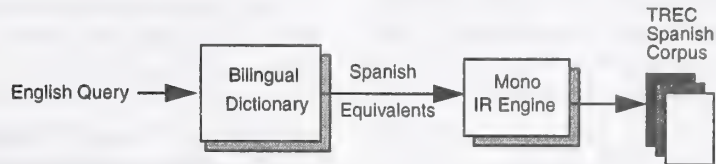


Figure 2: Replacing each English query term with all of its equivalents from the bilingual dictionary to form a new, ambiguous query.

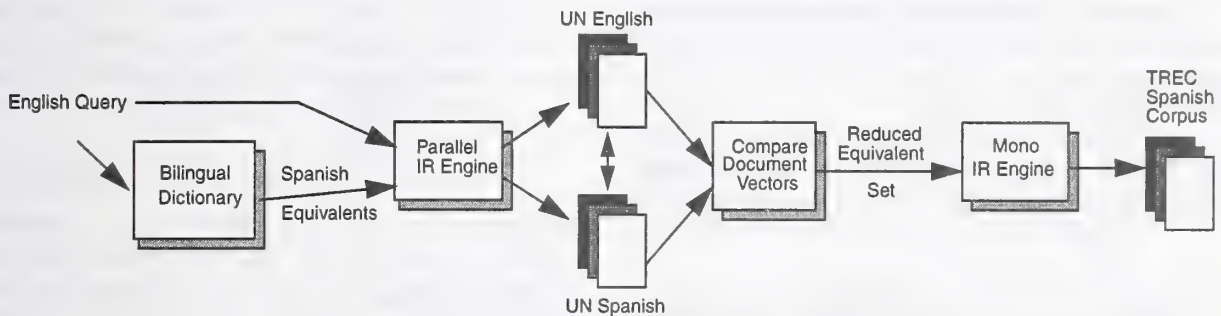


Figure 3: The corpus-based disambiguation method chooses among candidate equivalents for each term of the English query by measuring the similarity of the retrieval results for each equivalent to the English query on a parallel text retrieval task. The derived query is then submitted to the monolingual retrieval system.

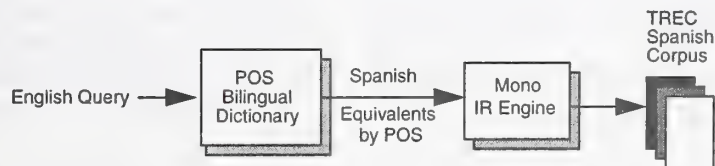


Figure 4: The POS disambiguation method chooses Spanish equivalents for each English query term by matching parts-of-speech in the bilingual corpus.

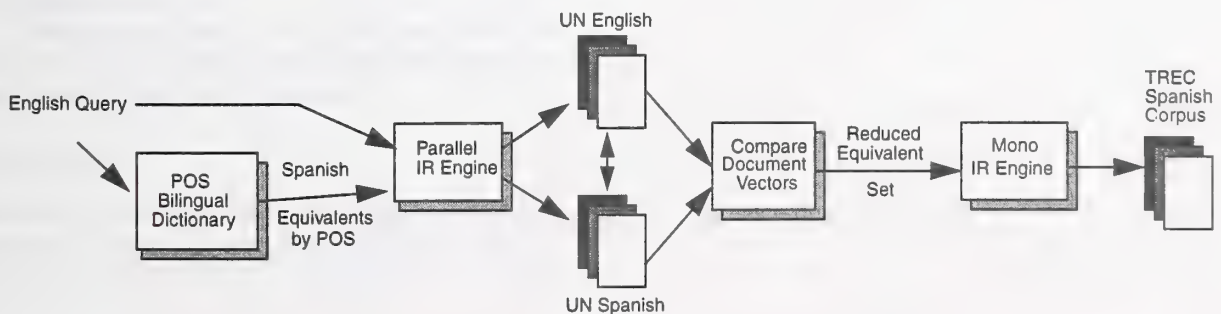


Figure 5: The combined approach uses the POS-disambiguated terms as input into the corpus-based disambiguation engine.

then chose only one of the candidate equivalents for each English query term to form the final query.

Monolingual and Cross-Language Retrieval Results

In order to evaluate the comparative performance of the monolingual system versus the disambiguated queries, TREC-4 and TREC-5 Spanish queries were used. CRL had previously provided English translations of the TREC-4 queries, so the English versions were already available and could be used alongside the Spanish versions. For the TREC-5 queries, NIST provided both English and Spanish versions of the queries.

In the discussion that follows, the monolingual Spanish results will be referred to as MONO4 and MONO5, the all-equivalent substitution approach as ALL4 and ALL5, the corpus-based disambiguation sets as CORP4 and CORP5, the POS approach as POS5, and the combined approach as BOTH5. The POS and BOTH approaches have not yet been tested on the TREC-4 query and document collections.

The pooled query-relevance judgements (qrels) from NIST were used to evaluate the system for several of these runs. It is possible that the stemming algorithm

that was used for Spanish might conflate Spanish terms in a manner not represented in the other systems, so the pooled qrels are probably not a perfect measure of the system's performance. There were no other options available prior to direct TREC evaluation, however. This applies to MONO4, CORP4, ALL4, POS5 and BOTH5, since each of these runs was not directly evaluated by NIST.

The performance of the three methods is shown in Table 1. The non-interpolated average precision values are listed by category.

Table 1 Average Precision For All Methods

<i>Method</i>	<i>PR (N)</i>
<i>MONO4</i>	0.1874
<i>MONO5</i>	0.2895
<i>ALL4</i>	0.0783
<i>ALL5</i>	0.1422
<i>CORP4</i>	0.1250
<i>CORP5</i>	0.1153
<i>POS5</i>	0.1949
<i>BOTH5</i>	0.2127

NMSU/CRL Trec5 Cross-language Retrieval Results

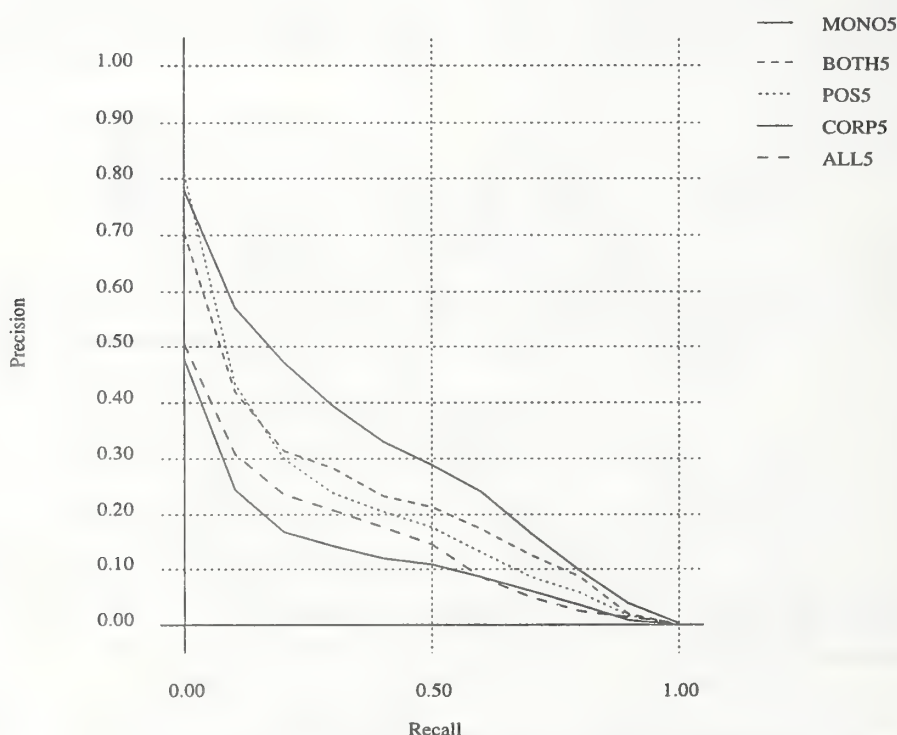


Figure 6: Precision Recall curves for Trec-5 Experiments

The complete Precision-Recall curves for MONO5, ALL5, CORP5 and BOTH5 are shown in Figure 6.

As can be seen, automatically translating a query into another language can have a substantial performance penalty, but by performing some simple disambiguation of query term equivalents, the penalty can be reduced substantially.

MONO4, ALL4 and CORPUS4 results are included because they show a slightly different pattern than the corresponding results for TREC-5. In the TREC-4 results, there was a clear performance gain that resulted from corpus-based disambiguation of the translation equivalents. For the TREC-5 results, however, corpus disambiguation decreased performance when used alone but was advantageous when combined with POS-based disambiguation. Exactly why this occurred is not altogether clear. It certainly must be due to substantial disambiguation errors being made over incorrect POS equivalents present in the TREC-5 query equivalent sets, but exactly why this query set performed so radically different from the TREC-4 set is not immediately apparent and will require further investigation.

Conclusions

Disambiguation of terms in an equivalent set supplied by a bilingual transfer dictionary can result in substantial improvements over most CLTR methods seen to date. The set of experiments presented in this paper provides a clear path to high-performance CLTR systems: combining POS-based disambiguation with corpus-based disambiguation for query translation. Further improvements are possible for the existing system, including accurate identification of phrases in the query that need specialized translations and, perhaps, interactive approaches to translating new terminology and acquiring lexicons for unfamiliar target languages.

Acknowledgements

The POS tagger used in this work was provided by MITRE Corporation. This work could not have been accomplished without it.

This research was funded under grant MDA 904-94-C-6153 of the US Department of Defense as part of the Tipster Reinvention Laboratory.

References

Church, K. W. and R.L. Mercer (1993) "Introduction to the Special Issue on Computational Linguistics Using Large Corpora," *Computational Linguistics*, 19:1. pp 1-24.

Davis, M. W. and T.E. Dunning (1995) "Query Transla-

tion Using Evolutionary Programming for Multi-Lingual Information Retrieval," In *Proceedings of the Fourth Annual Conference on Evolutionary Programming*, San Diego, Evolutionary Programming Society.

Davis, M. W., T. E. Dunning, and W. C. Ogden (1995) "Text Alignment in the Real World: Improving Alignments of Noisy Translations Using Common Lexical Features, String Matching Strategies and N-Gram Comparisons," In *Proceedings of the Conference of the European Chapter of the Association of Computational Linguistics*. University College Dublin. March.

Fogel, D. B. (1992), "A Brief History of Simulated Evolution," In *Proc. of the First Annual Conference on Evolutionary Programming*, ed. D.B. Fogel and J.W. Atmar, 1-16. San Diego: Evolutionary Programming Society.

Hull, D. and Grefenstette, G. (1996) "Experiments in Cross-linguistic Information Retrieval" in *SIGIR96*, August, Zurich, CH.

Landauer, T. K. and M. L. Littman (1990). "Fully Automatic Cross-Language Document Retrieval Using Latent Semantic Indexing," In *Proceedings of the 6th Conference of UW Centre for the New Oxford English Dictionary and Text Research*, 31-38. Waterloo.

Leacock, C., G. Towell, and E. Voorhees (1993) "Corpus-Based Statistical Sense Resolution" in *Proceedings of the Human Language Technology Workshop*, 260-265, Princeton, NJ. ARPA.

Salton, G. (1971) "Automatic Processing of Foreign Language Documents," in *The Smart Retrieval System*, ed. Salton, G., Prentice-Hall, Englewood Cliffs, NJ.

Wilks, Y. (1996) *Personal Communication*.



Experiments with TREC using the Open Text Livelink Engine

Larry Fitzpatrick, Mei Dent, Gary Promhouse
Open Text Corporation

Abstract

In **TREC-5** we baselined the Open Text Livelink Search Engine 6.1 and tested the use of a new automatic feedback technique against both the baseline and automatic top-document feedback. Baseline queries were created in a manner consistent with real users: small queries (average 5 word), created without benefit of query execution, manual feedback or external sources. The interesting results were that other similar queries used as a source of new evidence for automatic query augmentation (*feed-forward* returned a 38% average precision improvement over the baseline, a 12% average precision improvement over automatic top-document feedback, a 6% improvement in top-document feedback (at the 5 and 10 document levels), and was amenable to thresholding for optimal application of the technique. Automatic top-document feedback yielded nominal improvements and hurt top-document precision, which is consistent with the literature. Attempts to use the embedded document structure to improve search results showed no improvements, despite subjective

judgments in other domains that this can be worthwhile.

1 Introduction

This is Open Text's first participation in a **TREC** conference, even though the history of the Livelink technology extends back into the middle 1980's. We are participating as a Category C system in the *ad hoc* query track using the *off-the-shelf* Open Text Livelink Search Engine *Release 6.1*.

Our objectives in this participation were twofold. First, we wanted to test various subjective observations that the use of document structures in relevance scoring can improve retrieval effectiveness. Second, we evaluated several automatic query expansion techniques.

Early explorations in this **TREC** effort attempted to characterize the suitability of the **TREC** data to several structure weighting techniques. The text algebra that is the query language for the Livelink engine [8] integrates low-level relevance score composition functions with rich structure handling primitives. We

have employed these features in a number of installations to customize scoring functions to the data and application yielding noticeable improvements in relevance scoring in limited subjective evaluations.

The experience in online environments, of others (e.g., [10] with Okapi) and ourselves (e.g., the *Open Text Index*), shows that users generally make little investment in searching (e.g., supply few query words, provide limited feedback) yet do notice the effectiveness (or ineffectiveness) of their searches. To address this, we have explored techniques that automatically augment user queries. We compared two separate automatic augmentation techniques using the TREC data: top-document feedback and feed-forward.

To facilitate useful comparisons, first we manually constructed queries with nominal intellectual input and generated runs from them to serve as a baseline (OTC-1). Then, as a basis for comparison against the baseline and against other augmentation techniques we measured automatic augmentation by the top ranked documents (*top-document feedback*). This technique is described by [2]. The run describing it is OTC-2.

Third, we explored the idea that a query is really an event that should not be decoupled from the user issuing it. We simulated the existence of a *user context* and measured augmentation by this private history. This is similar to the query augmentation described in [4]. The benefits to this technique are seen in a client-server environment where user context is readily (and privately) available and the

processing burden imposed by augmentation is distributed to the client. The run describing *feed-forward* is OTC-3.

2 System Description

2.1 Search software

The roots of the Livelink engine are in the *New Oxford English Dictionary Project* undertaken by the University of Waterloo [1]. The original engine was built to handle the demanding query requirements of structured data as found on this project. Salient to this discussion is that the Livelink query language is based on a *text algebra*, PAT, [8] and that the data model imposes no fixed notion of a *retrieval unit* (or *document*) at index time. In fact, the notion of document is a flexible in that the retrieval unit is defined at query time. Instead, the indexing process defines *regions* over the text which may correspond to words, sentences, text fragments, fields, documents, aggregations of documents, etc. Regions may be generated from the data, externally supplied metadata, or may be added at runtime by query or by extension.

The statistical scoring capabilities of the Livelink engine exist as primitives in the *text algebra* and incorporate the age-old probabilistic components: global term weights, document term frequency, document length, externally supplied term weights, and a full complement of score combinators. These values and operators can be computed and applied over any regions defined in a database, not just what

one normally thinks of as *the document*, or *retrieval unit*. Also unlike most systems, score generation for user queries is flexibly defined by the mapping of the user query language onto the text algebra primitives.

The upshot is that by custom combination of the region algebra with the ranking expressions, great flexibility can be obtained in defining how a retrieved item is to be scored. Because of these factors, we frequently encode application and data specific knowledge into scoring functions.

2.2 Test environment

The system description of the hardware used for index building and query evaluation is as follows:

Processor: DEC ALPHA server 2100

CPU Clock: 200 MHz

RAM: 128M available

Hard Disk: 74 GB hard disk

The memory requirements of the Livelink engine are configured by the administrator. To build the **TREC** database, 60 Mbytes of memory was used. This machine was both a development machine and a file server, and was rarely dedicated solely to **TREC** work.

3 Description of Experiments

Our efforts were directed at evaluating techniques for improving retrieval effec-

tiveness. The first was the use of document structure. The second was the use of two automatic query augmentation techniques. We generated 3 runs that we will discuss, named **OTC-1**, **OTC-2**, and **OTC-3**. Here we provide a brief overview of the approach, and in the following sections describe each set of experiments in more detail. However, we first need to point out an embarrassing error.

The runs that we completed before the submission deadline were executed against two-thirds of the database, and resulted in loss of access to 519 of the 5524 total relevant documents. We discovered this when we received relevance judgments on the 16th of October and recognized that the recall-precision numbers were approximately 25% below the test runs we had done internally against the **TREC-3** collection. The affected runs are **colm1** and **colm4** and are replaced by **OTC-1** and **OTC-3**. Henceforth, we refer to runs **OTC-1** and **OTC-3** for all comparisons, as **colm1** and **colm4** are useless. Figure 8 and Figure 10 are replacement recall-precision graphs for the official ones in the appendix run using the full data set.

It is well known in operational environments that users do not provide very many query terms. On the Open Text Index (a publicly available database created by crawling the WWW), accumulated statistics show the average number of query terms to be around 2. As the volume of data made available by search engines increases, we have to ask: *How will 1 or 2 query words effectively partition 10's of GB of data?* To counteract this effect,

we have been exploring models of query augmentation that rely on implicitly derived evidence, as opposed to explicit feedback.¹ First, we evaluate the use of automatic top-document feedback. While not new, it is already part of the product and it provides a basis for comparison with other augmentation techniques and with the past literature on this method.

Second, we explore automatic query building by *feed-forward* augmentation. The premise of *feed-forward* is the notion that queries are not isolated one-shot events, but are often framed by the longer-term information needs of a user or community of users. We simulate a long-term user need and use this as the source of data for query augmentation. A number of related efforts over the last several years have looked at using past user behavior to improve information discovery activities, mostly in filtering applications ([6], [5]), but also in searching ([4]).

Each of our runs is described in Figure 1, and then explained in more detail in the following sections.

3.1 Baseline - OTC-1

The generation of the baseline run consisted of index building, query creation and query execution.

¹Not that we're opposed to explicit feedback methods. In fact we support those as well, we just do not describe them here.

3.1.1 Index building

The index contained a total of 1,371,876 documents. The documents were not reformatted during index building. The Livelink engine supports the logical addition of *metadata* to each document. We added metadata regions to identify which sub-collection each document came from, and which TREC CD it came from. This allowed us to divide the data into sub-collections at query time without having to build separate sub-databases.

Each SGML tag in the data was used to create a region for that tag name. In addition, certain SGML tags with common meaning were grouped under one region name. For example, the SGML tags in the data that identify a headline were <HEAD>, <HL>, <HEADLINE>, depending on the collection, and during index building stage we grouped these tags into the region **Headline**.² This enabled us to search independently for <HEAD> or <HL>, or alternatively any of the headline forms using **Headline**.

The TREC database was built in 4 hours.

3.1.2 Query creation

TREC-5 queries were manually created. All topic fields were used to create the query, although emphasis was placed on the **Description** field. We followed some rough guidelines to approximate real users' queries: (a) limit query operators to

²The documents were not changed in any way to do this.

stem expansion, either of the *or* operators or the *near* operator (which we used sparingly), (b) do not execute the queries or use TREC data in any way, (c) do not look at external resources for term generation, e.g., dictionaries or thesauri, and (d) keep queries reasonably small. On average about five terms per query resulted.

The average time to create a query for a topic was on the order of several minutes, but varied greatly. The total resource used was around one person-day.

3.1.3 Query execution

The baseline queries were run on the same machine as the indexing machine. Absolute CPU time consumed was under 60 seconds for all queries. Total elapsed time per query is under 1 minute, affected by factors such as network load, other development activities, etc.³

3.2 Using structure

A region in Livelink is a span of text in the database, defined by start and end offsets. Regions generally are used to represent portions of the text that have something in common. For example, a newspaper may be composed of many *story* regions, each of which contains a *title* region, a *headline* region, and a *body* region. The Livelink engine offers great flexibility in both the creation of these regions at build time or

at run time, as well as the use of these regions to affect the rankings of documents.

We have used the region structure in a number of applications to improve perceived retrieval effectiveness. For example, on the Open Text Index, query terms matching text from the automatically generated summary region were weighted more than matches from the raw text, and query terms that appear to be site names were weighted more heavily when matched out of the URL region.

We ran a number of tests on structure weighting with the TREC-3 collection and had intended to test this capability against TREC-5. However, the results of the experiments were not significant enough to warrant any TREC-5-time. But, here is a brief overview of what we tried and observed and we will not discuss it further.

Experiments were run to see if weighting regions differently during ranking of the documents had a measurable effect. Nothing more than nominal differences were observed. We attribute this to several factors. First, the TREC record structure is very limited in semantics, with *headline* the only widely used and truly notable feature. We posit that using this structure had no effect because many of the documents were newspaper articles and their construction of headlines is often obtuse. Second, the different TREC subcollections actually have different structure, which makes the problem more difficult. We did not try to compensate for this, outside of the mention of normalizing those things that were obvious.

³The machine was actively used as a departmental server during this effort, though we would have enjoyed having an Alpha 2100 all to ourselves!

We also partitioned long documents into sections to see if it improved retrieval effectiveness. We tried various partition sizes and various score combination functions. The partition size ranged from 50 to 200 words and the overlap ranged over 0, 10, 20, and 50. We noticed that varying the size of the artificial regions that fragmented a larger retrieval unit into smaller ones did not change the overall performance significantly, as observed by [11]. Also, we tried several score combination methods for use with fragments including maximum, average, median, and total. The best performer was the maximum score of the individual fragments.

3.3 Top document feedback as augmentation - OTC-2

We tested automatic top-document feedback using the *find similar pages* capability of the Livelink engine. This provided a measurement of the effectiveness of this technique with the Livelink engine as compared with research reported elsewhere ([2]). Also, it allowed comparison with the other automatic augmentation technique we tested, *feed-forward*.

This method was virtually identical to what [2] described. The baseline query was executed against the collection and the result list obtained. The top several documents were selected and terms were extracted from them. The top terms were ordered by importance using a function that took into account global and local term weights in addition to lexical similarity with other selected words. The top

several terms were added to the original query and the query was re-executed. The parameters to this method were number of top documents to use and the number of top terms to select from the top documents and were tuned previously using TREC-3 data.

Top-document feedback imposed an additional query processing load due to key term extraction and then re-execution with the augmented query. The augmented query took slightly longer than the base query to evaluate due to the additional terms (about 2x the original). The cost of key term extraction was nominal.

3.4 User context as feed-forward augmentation - OTC-3

We make the hypothesis that in many situations, search can be improved if treated more like a filtering activity. Filtering satisfies chronic information needs and so IR efforts on this problem use past performance to improve later activity. A search, on the other hand, has almost always been treated as an isolated event that satisfies an acute need. If we accept that a user's motivation to search is frequently consistent with longer term information needs, then his past information consuming habits should be predictive of relevance in a given search.

The idea that a user's past interests can be used to modulate a query's ranking order was explored in the Okapi project [4]. In this effort, a user's past documents provided *context* used to break ties in relevance scores of the query result list. Many

other efforts have explored the use of user's past efforts to facilitate future discovery efforts ([5], [6] to name only two). Here, we explored this notion with the Livelink engine and a query's background context.

To test this hypothesis, we wanted a history for a *user*. TREC does not provide this directly. However, we simulated a user's past interests, in the context of a single query, and used this information as the basis for *feed-forward* query augmentation. The source of this *past context* was other TREC queries. The basic *feed-forward* approach is as follows:

- Simulate a background context for each query, consisting of several hundred documents. We call this the *affinity corpus* for the query.
- Execute the query against this affinity corpus and select the top n documents.
- If the top documents exceed a similarity threshold to the query, select the m most representative terms from these documents to augment the query.
- Execute the augmented query against the full corpus and keep this result list.

The only information available to create an affinity corpus was the complete body of TREC queries. We used a complement of all 50 TREC-3, 10 TREC-4, and all 50 TREC-5 queries executed against the TREC-5 data as candidates for similar

queries. To create an affinity corpus for each query, we selected the top 100 documents from each of the 3 most related queries. We compared queries by result-list ([7]) using a weighted Jaccard score that counted the overlap between the top 100 documents of each result list.⁴

The assumption was that a user with a given query need may have explored that need before and would have seen several documents related to his present need and these documents would be in his affinity corpus. We experimented with several different numbers for the top documents and top terms selected from the affinity corpus to augment the baseline query.

Since some queries had no queries like them, we applied a threshold to the augmentation. If the affinity corpus had low similarity to the query, as measured by a low query-query similarity score, we did not augment the query with terms from the affinity corpus. We experimented with several threshold values and settled on one that allowed about half of the queries to be augmented.

It took about a minute of elapsed time to create the affinity corpus for each query from the 109 other query results. To select the background documents and create a *feed-forward* augmented query added less than a second to each query. Remem-

⁴Initially, we tried to find related queries by manually grouping topics based on subjective judgments of relatedness. We found this to be a painful classification exercise, fraught with ambiguity. Fundamentally, it didn't take into account the underlying data. We abandoned this effort reasonably quickly.

ber, the affinity corpus was only 300 documents and this effort could take place at a client machine, thus adding no server burden for creating the augmentation. The augmented queries took roughly a factor of two longer to run than the base queries due to more terms. Finally, for queries that did not have a sufficiently strong affinity corpus, i.e., did not exceed the threshold, we just used the base query.

4 Discussion of Results

We executed three separate runs: a baseline run (OTC-1), an automatic top-document feedback run (OTC-2), and the automatic feed-forward using past queries as evidence (OTC-3).

4.1 Result analysis for baseline run - OTC-1

The baseline run measured the effectiveness of the manually constructed queries with the default relevance ranking algorithm. This run was very close to the median run for TREC-5, on average. The total number of relevant documents retrieved was 2741, and the average precision was 21.33%.

Figure 2 shows the number of topics that performed better or worse than TREC-5 median, and the percentage of increase or decrease in average precision. The median difference is around +5%. Note that when a baseline query did worse than the TREC-5 median, it was likely to do very much worse. Out of the 16 topics with an average precision worse than

the median by more than 10%, we were later able to improve half of them to outperform the median by adding only one or two query words, by applying simple query refinement using only a few minutes per query. Clearly, we were hurt by not using manual feedback or extra-data resources when building these queries.

4.2 Result analysis for top-document feedback run - OTC-2

This run tested query augmentation by automatic top-document feedback. The engine's *find similar pages* feature generated the augmented query terms.

Both average precision and the total number of documents retrieved increased over the baseline. The total number of relevant documents retrieved was 3112, 13.5% above our baseline, OTC-1. The average precision was 22.91%, about 7.5% above OTC-1. The results are compared in Table 1.

Figure 3 shows the number of topics that performed better or worse than our baseline run (OTC-1), and the percentage change in average precision. Interestingly, comparing on average precision, when top-document feedback helped, it tended to help a lot and when it hurt, it only hurt a little (with one exception). Close examination of the results showed that when the top documents used for feedback were all relevant, the greatest improvement of average precision was achieved. When none of the top documents used for feedback were relevant, moderate improvement of

precision	top-document		feed-forward		baseline	
	all queries	augmented	all queries	augmented	all queries	augmented
average	0.2291	0.2232	0.2465	0.2606	0.2133	0.1884
exact	0.2736	0.2629	0.3095	0.3112	0.2693	0.2238

Table 1: Average and exact precision for baseline, top-document feedback and feed-forward methods.

average precision was achieved, partially because the top document precision in the baseline run was so poor. When the top documents used for feedback were a mixture of relevant and non-relevant documents, results were mixed.

4.3 Result analysis for feed-forward run - OTC-3

The result of the OTC-3 run showed an increase over the baseline run in both average precision and the total number of documents retrieved. The total number of relevant documents retrieved was 2877, about 5% over our baseline, OTC-1. The average precision was 24.65%, just over 15% above OTC-1.

Figure 6 shows the number of topics that performed better or worse than our baseline run, and the percentage change in average precision. The median difference was over +15%. When a proper threshold was applied, great improvement in average precision was achieved.

The application of feed-forward augmentation for a query was subject to a threshold. Of the 50 total queries, only 23 queries exceeded the threshold. For the 27

queries that could not be augmented, the baseline run (OTC-1) was used.

When we compared the feed-forward method against the baseline and against the top-document feedback method, the improvement is much more marked. Table 1 lists the average precision and recall for just the *augmented* as well.

Comparing just augmented queries, feed-forward showed a 38% improvement in average precision over the baseline which constituted a 12% improvement over top-document feedback. Figure 4 shows the recall-precision curves for all queries and Figure 5 shows the recall-precision curves for the augmented-only queries. The separation for all recall-precision points (except the 0th) is more pronounced for augmented-only queries. The ability to change the threshold to trade greater or lower levels of improvement against lesser or greater numbers of queries against which to apply it seems powerful.

Conventional wisdom holds that automatic feedback hurts top-document precision. Figure 7 shows that while top-document feedback did hurt top-document precision, the feed-forward method actu-

ally improved it.

5 Conclusion

As a first time participant, we certainly learned a lot about how to work effectively within the TREC evaluation framework. Except for the successful evaluation of a new automatic feedback technique, unfortunately, much of that knowledge will have to be applied to the next TREC and is not evident here.

After correcting for the error in which we failed to index a substantial portion of the relevance judgments for our official runs, our baseline performance was slightly better than the TREC median for the manual ad hoc category. Our misjudgment was in the method we used for building queries from TREC topics. It did not provide a favorable base for comparison against other systems, as it was neither fully manual nor fully automatic.

To create queries from topics, we read the topics and in a few minutes composed, on average, five word queries and used this as a baseline. This is consistent with what real users do. We did not consult external sources, the data itself, or use feedback in any way. However, we think we would have been better served to either fully automate the query creation step and participate in the automatic ad hoc category, or to rely on feedback and consultation of knowledge bases to refine the query before settling on baseline queries for the manual run - live and learn. Most systems in the manual category appear to have

invested substantial effort in the latter. That notwithstanding, we demonstrated what we believe to be important advances in the application of automatic feedback methods. But, why automatic feedback at all when the results in the literature have been ho hum?

Our past experience indicates that end users in production search environments do not provide queries sufficiently rich to be effective for very large data collections. As collection sizes continue to increase, and as more users engage in searching, this situation promises to get even worse. One approach to improve search effectiveness for end users is to engage them in activities that tease out more information to help them formulate better queries. A complementary, not incompatible, approach is for the search environment to assist automatically, using whatever it can learn about past user, group, and system behavior to improve search effectiveness. We believe there exists much untapped evidence in end-to-end query systems.

In this TREC-5 effort, we've shown some very positive results using implicitly derived information to help search effectiveness. Automatic feed-forward, using past queries as additional evidence, was superior to automatic top document feedback and actually improved top document precision.

When we compared only the 23 queries for which feed-forward augmentation was performed, the results were surprisingly good. Average precision for feed-forward was 38% over the baseline and 12% better than top-document feedback. Moreover,

feed-forward augmentation improved the top 5 and top 10 document precision levels by about 6%. Consistent with reports elsewhere ([2]), automatic top-document feedback yielded nominal improvements over the baseline of 7.5% in average precision and 13.5% in recall and hurt top-document precision by about 5%.

The use of an empirically derived threshold for determining when to apply feed-forward definitely helped. Performance of feed-forward applied indiscriminately was nominally the same as the top-document feedback method. The fact that the feed-forward method using past queries as additional evidence could be easily subjected to a threshold is substantially in its favor. We do not think, however, that we have exhausted the possibilities.

We think that for many cases, a search is not an isolated event, yet this is how virtually all systems treat them. Of necessity, our efforts here took a system-centric view of looking for additional evidence⁵ to help augment queries, because TREC does not provide any notion of a user. We most certainly have not exhausted system-centric sources of additional evidence. For example, we compared queries to past queries by result-set overlap. In this comparison, we derived judgments about which parts of the result list were likely to contain relevant documents, and so were weighted more heavily. We completely ignored in this evaluation (because we had to), which documents from past queries may have

been viewed, selected, saved or printed by the original querier, thereby indicating higher probability of relevance ([6] studied several such factors applied in a non-searching context).

If system-side evidence can help automatically improve queries, and we think we have shown that it can, then surely user-specific evidence collection, located client-side so as not to impose a load on the server, appears even more promising. By simple observation of user behavior we note that user-generated queries often overlap past queries issued by the very same user. If a single query can be helped by other users' past query efforts, wouldn't the same user's past queries perform at least as well? Even more likely as a source of evidence than repeat queries is the probability that the user has read an electronic document related to the subject he is querying, whether it was email, web-browser, or other. What of the documents from those similar past queries that that the same user viewed, saved, printed, etc.? Why not what he has written? Can we identify evidence in a user's past information consuming habits to help future information seeking? We think so.

References

- [1] The New Oxford English Dictionary Project at the University of Waterloo, D.L. Berg, G.H. Gonnet, F.Wm. Tompa. Technical Report OED-88-01.

⁵other users' past queries

- [2] C. Buckley, G. Salton, J. Allan, A. Singhal, Automatic Query Expansion Using SMART: **TREC 3**, Proceedings of Third Text Retrieval conference (**TREC-3**), NIST Special Publication 500-225, 1995.
- [3] An Algebra for Structured Text Search and a Framework for Its Implementation, Charles Clarke, G. Cormack, and F. Burkowski, The Computer Journal, V38.No.1 1995.
- [4] Goker, A. and McCluskey, T. L. Incremental learning in a probabilistic information retrieval system. In: L.A. Birnbaum and G.C. Collins and (eds). Machine Learning. Proceedings of the Eighth International Workshop on Machine Learning (ML91), Evanston, IL, USA, June 1991. San Mateo, CA: Morgan Kaufmann, 1991. 255-9.
- [5] Pattie Maes, Agents that Reduce Work and Information Overload, Communications of the ACM, Vol. 37, No.7, July 1994.
- [6] M. Morita and Y. Shinoda: Information Filtering Based on User Behavior Monitoring and Best Match Text Retrieval, Proc. of the Seventeenth International ACM-SIGIR Conference on Research and Development on Information Retrieval, 272-281, Dublin, Ireland, July 1994.
- [7] Vijay V. Raghavan, Hayri Sever, On the Reuse of Past Optimal Queries, Proc. of the Eighteenth International ACM-SIGIR Conference on Research and Development in Information Retrieval, Seattle, Washington, July 1995.
- [8] Pat expressions - An algebra for text search, Airi Salminen, Frank Tompa. Technical Report OED-92-02, 1992.
- [9] An Overview of Waterloo's Database Software for the OED, Frank Wm. Tompa. Technical Report OED-92-01, 1992.
- [10] Walker, S. and de Vere, R. Improving subject retrieval in online catalogues: 2. Relevance feedback and query expansion. London: British Library, 1990. (British Library Research Paper 72).
- [11] Ross Wilkinson, Effective Retrieval of Structured Documents, In the ACM-SIGIR Conference on Research and Development in Information Retrieval, 1994.

A Appendix

Run	Description
colm1	Baseline run, manually created queries, partial database. Submitted, garbage run. Ignored.
colm4	Automatic expansion of query. Submitted, garbage run. Ignored.
OTC-1	Baseline run, manually created queries. Used as benchmark to compare other runs.
OTC-2	Automatic expansion of query by top ranked document feedback.
OTC-3	Feed-forward augmentation. Automatic expansion using simulated user context-documents that a user may have seen in the past.

Figure 1: Name and description of each run.

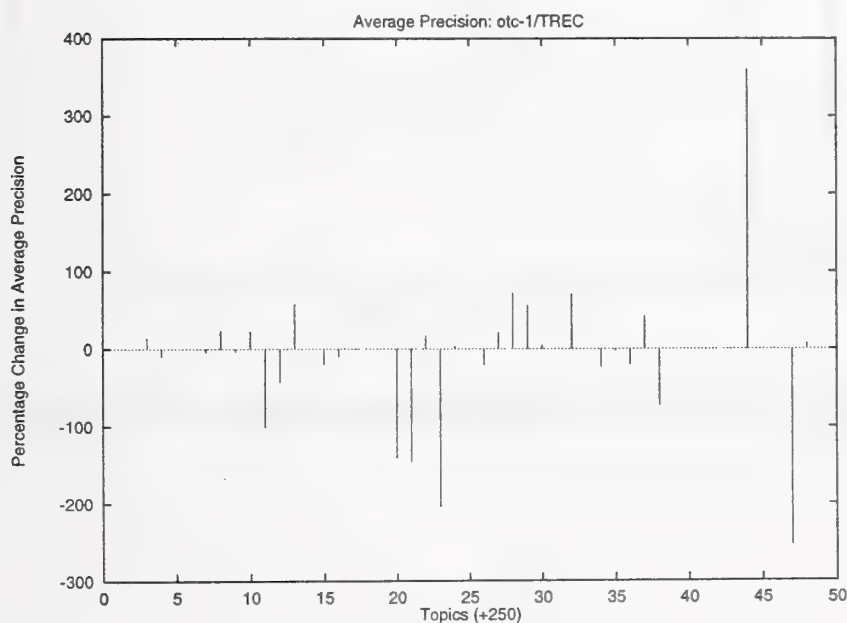


Figure 2: Percentage change in average precision for each topic: OTC-1 vs. TREC-5 median.

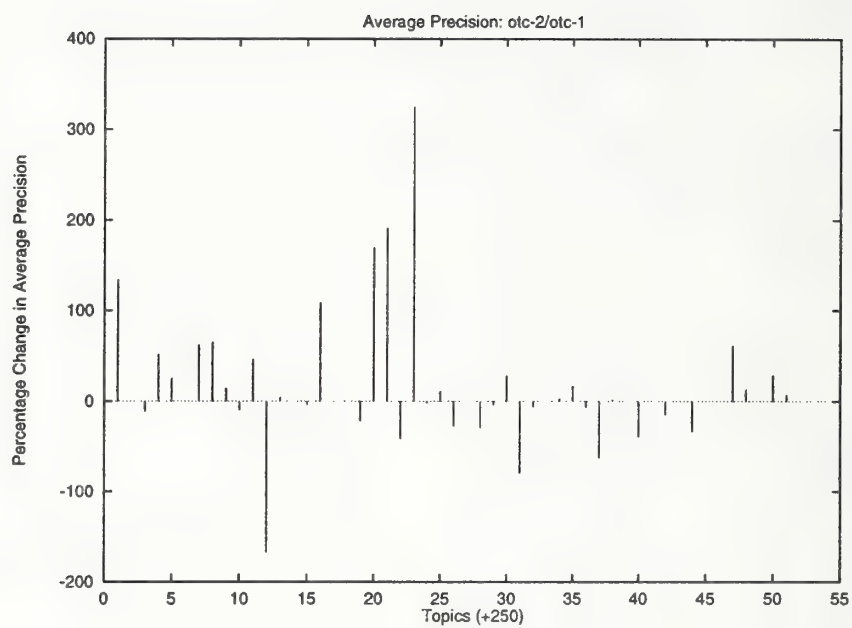


Figure 3: Percentage change in average precision for each topic: OTC-2 vs. OTC-1.

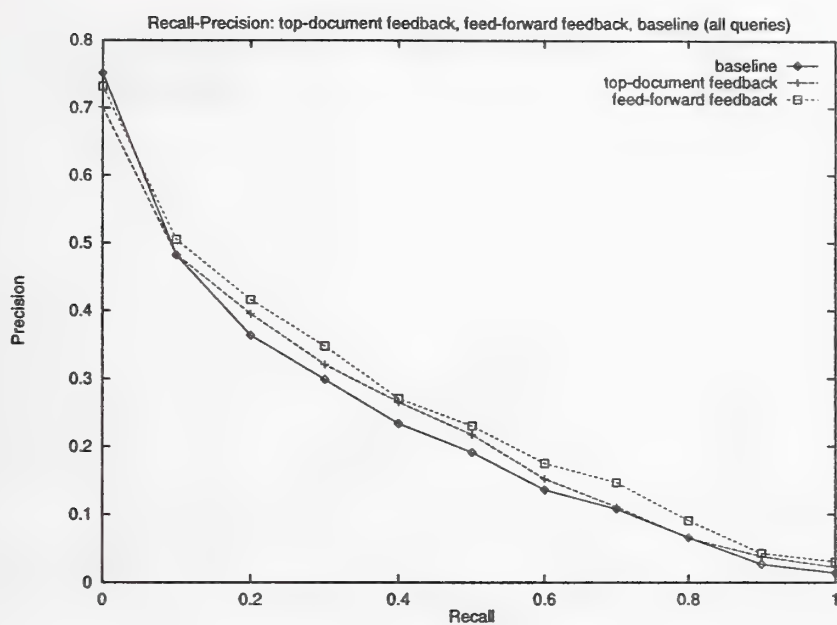


Figure 4: Recall-Precision for baseline, top-document and feed-forward feedback methods for all queries.

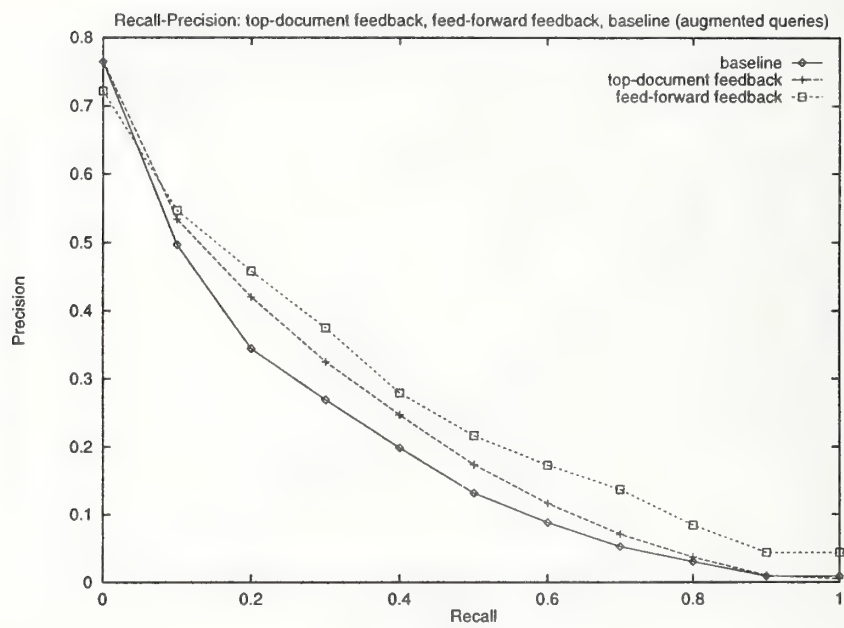


Figure 5: Recall-Precision for baseline, top-document and feed-forward feedback methods for augmented queries only.

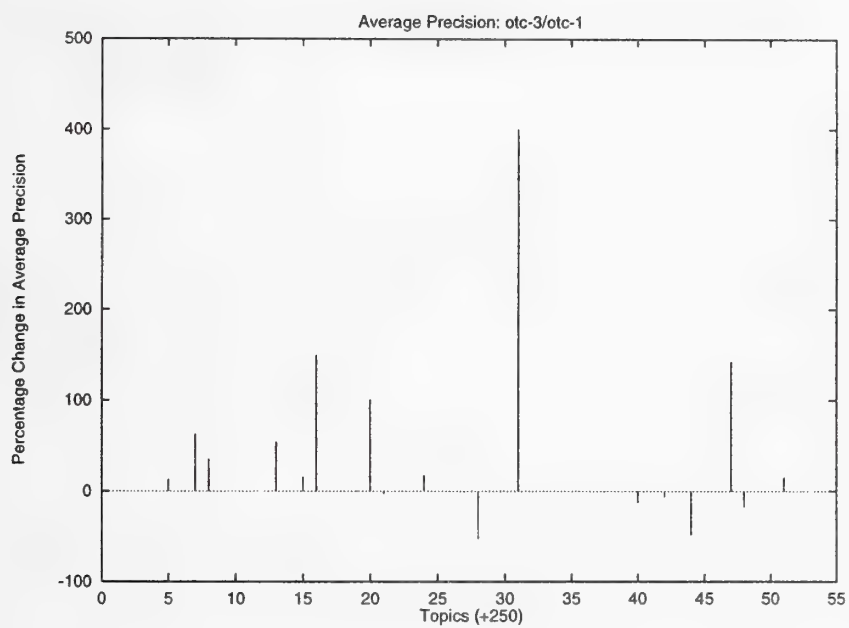


Figure 6: Percentage change in average precision for each topic: OTC-3 vs. OTC-1.

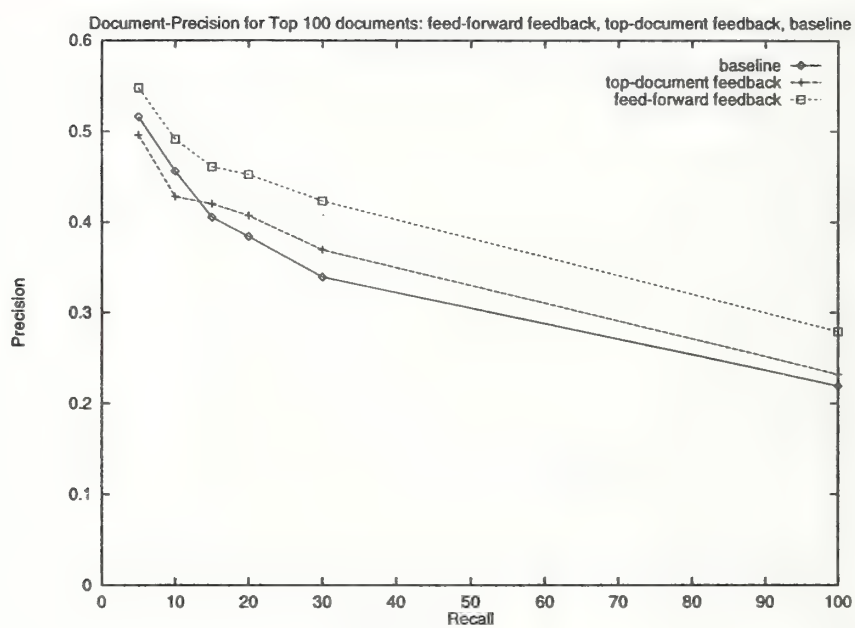


Figure 7: Document-Precision at top 100 documents for baseline, top-document feedback, and feed-forward methods.

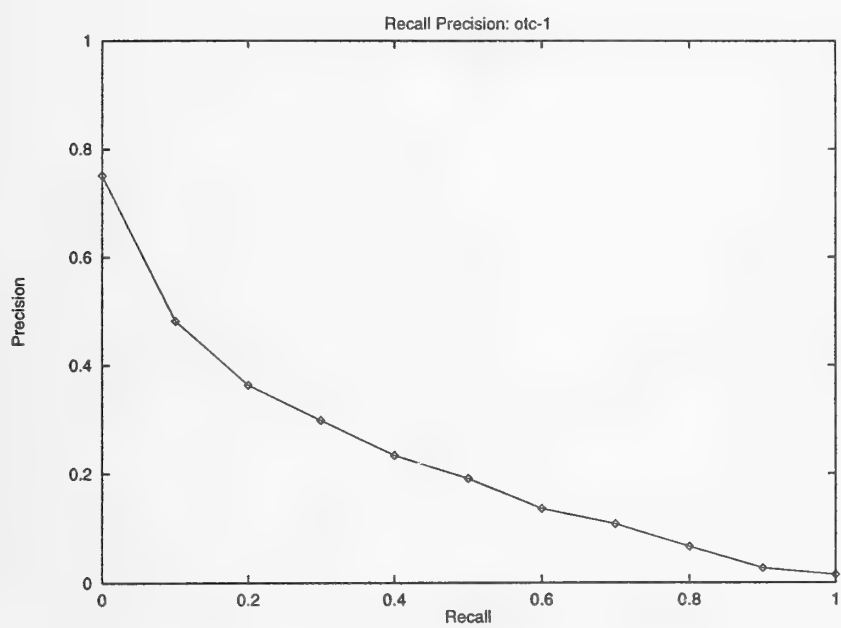


Figure 8: Recall precision for OTC-1.

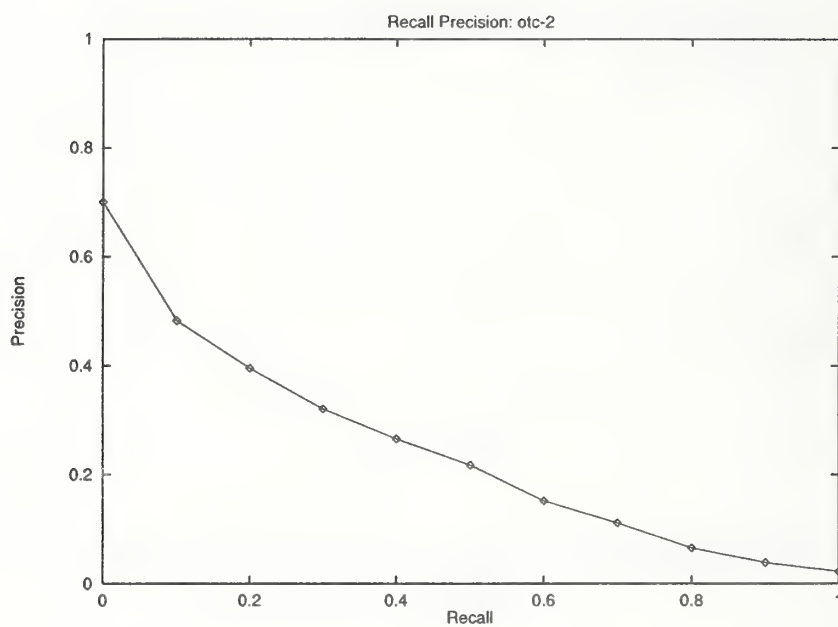


Figure 9: Recall precision for OTC-2.

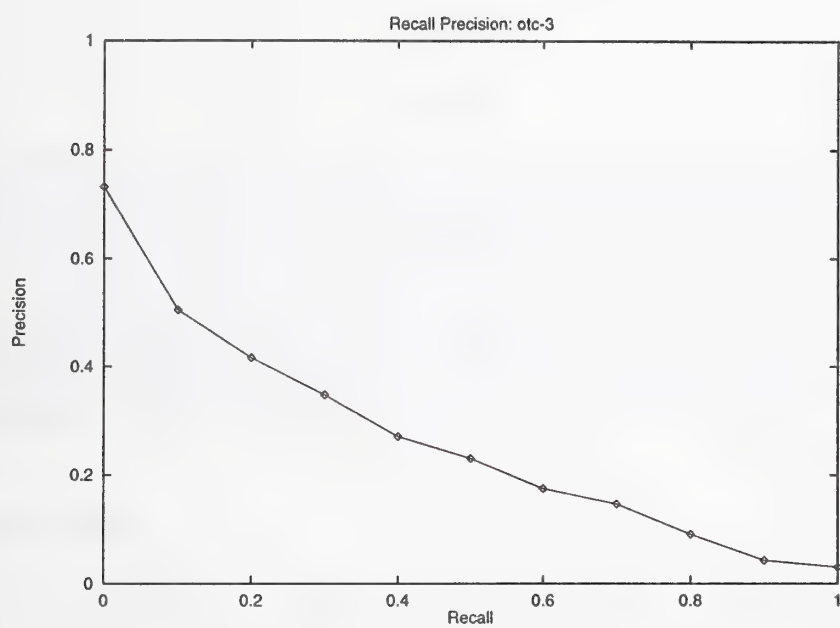


Figure 10: Recall precision for OTC-3.



Data Fusion of Machine-Learning Methods for the TREC5 Routing Task (and other work)

Kwong Bor Ng* David Loewenstern† Chumki Basu‡
Haym Hirsh§ Paul B. Kantor¶

Rutgers University
New Brunswick, NJ 08903

Abstract

The goal of the document routing task is to extrapolate from documents judged relevant or irrelevant for each of a set of topics accurate procedures for assessing the relevance of future documents for each topic. Rather than viewing different approaches to this problem as “winner-takes-all” competitors, we view them as potentially complementary methods, each exploiting different sources of information. This paper describes two quite different machine-learning approaches to the document routing task, and two approaches to combining their results to perform relevance assessments on new documents. We also describe an approach to the the confusion task based on n-grams that allow approximate matches.

1 Introduction

In the TREC routing task there is a large collection of documents, some of which have been assessed for relevance to each of a number of topics. This makes it possible to approach the routing task as a machine-learning problem — to extrapolate from the topics and evaluated documents some method for accurately assessing the relevance of future documents to each topic. However, applying a machine-learning method to this problem begins with the need to define the attributes that will be used to characterize the documents, over which the machine-learning methods will operate.

Historically, since the work of H.P. Luhn [Luhn, 1958] word tokens (or their morphological stems) have been the preferred attributes for characterizing documents. Although commercial retrieval systems also make use of word position (as does the more sophisticated Inquiry system [Turtle and Croft, 1991]), many others use a simple vector representation in which each document is

*APLab, SCILS, kbng@kantor.rutgers.edu.

†Computer Science Department, and Bell Laboratories, Room 15F315, 67 Whippany Road, Whippany, NJ 07981, loewenst@paul.rutgers.edu.

‡Computer Science Department and Bellcore, 445 South Street, Morristown, NJ 07962, cbasu@paul.rutgers.edu.

§Computer Science Department, hirsh@cs.rutgers.edu.

¶APLab, SCILS, kantor@scils.rutgers.edu.

represented by the frequencies with which terms (or stems) appear in it. More recently, though, some success has been claimed for another approach, using overlapping sliding windows of length 5 [Damashek, 1995] or 4 [Grossman *et al.*, 1996].

In what follows we distinguish between the representation itself (which is a mapping of documents into some space of attributes whose structure may be quite complex), and a “scheme”, which is a method for assessing the relevance of documents for a topic, usually based upon a particular document representation. Although one scheme may be uniformly more effective than another (with respect to both corpora and information requests), sometimes there are systematic dependencies of the effectiveness of the representation on the corpus, the information need, or both, and in some situations the two may vary in effectiveness in ways that are not systematically determined by any known variables.

A key idea in this work is that if one scheme is sometimes superior to and sometimes inferior to a second, then it may be possible to combine the two to achieve better performance than is possible with either in isolation. We have explored two specific methods for such “data fusion” in our TREC5 work. The first is to find, for each topic in isolation, which scheme works best, and use it to perform the routing task for that specific topic. We call this approach “scheme-per-topic”, abbreviated *SPT*, since each topic is assigned its own single scheme. Our second approach is to assess the relevance of a document as some combination of the relevance assessments of the different schemes. We call this second approach the “global” method, abbreviated *GLOB*, since our specific combination method uses a single rule of combination across all topics.¹

One would expect data fusion to be more successful as the difference between the schemes increases. In TREC4 we found that term-based and 5-gram-based representations were not sufficiently different to support interesting fusion experiments [Ng and Kantor, 1996]. We are now exploring data fusion on two schemes that appear to differ more significantly from one another. Although our preliminary work (described in the Workshop Paper) examined four different schemes, the fusion experiments reported here focus on only the two we were able to complete in time for the TREC5 conference.

The first of these schemes (called *KB*) is term-based, implemented using *mgquery* [Witten *et al.*, 1994], with a small number of query terms selected according to their individual ability to discriminate the union of all relevant documents from the union of all not-relevant documents. The second (called *DL*) estimates the compressibility of each document in terms of the dictionary built from compressing the topic statement plus the corpus of judged-relevant documents.

Unfortunately, our implementation of the *DL* and *SPT* methods had bugs not discovered until after our official submission was made. The results reported here represented an exploration of the *GLOB* method, albeit when given inferior data.

We begin this paper with a description of these two machine-learning schemes for the routing task. We then present the two fusion methods we used in this work. Results comparing our two fully implemented (and in one case buggy) routing schemes as well as the *GLOB* fusion method are then presented. We next present some results on the effectiveness of the *DL* method expressed in terms of a signal detection approach. These results help us to understand why the method was not more effective as implemented, and to lay the groundwork for possible modifications that would make it more effective in the IR routing situation. Finally, we discuss some additional work we performed

¹This is in contrast to adding an additional layer in which a different rule of combination is formed for each topic [Bartell *et al.*, 1994].

on the TREC5 confusion track. We conclude the paper with some final remarks concerning the philosophy underlying our collection of TREC5 efforts.

2 Term-Based Relevance Assessment: Scheme *KB*

To select the terms used in term-based representations of documents, we considered the union of the texts of all the known relevant documents (P), and the union of the texts of all the known non-relevant documents (N). The appearance of each term (stemmed as in the General Electric-Rutgers collaboration, reported elsewhere) was assumed to follow a Poisson distribution, and the parameter of that distribution was estimated (computed on the union of all the texts of a given type). The ratio of the estimated parameters — π for the positive text P and ν for the negative text N — was used to rank the stems for their ability to discriminate between the two texts. To avoid the occurrence of zeros and infinities we used an approximate confidence interval estimator of the Poisson parameter. In the positive texts, the parameter was estimated to be at the lower edge of a confidence interval. For the negative texts, the parameter was estimated to be at the upper edge of the confidence interval.

More specifically, we chose π (resp. ν) to be that point below (resp. above) the unique value of the Poisson parameter that makes the likelihood of the observed value a maximum (that is, $\lambda = k$), and such that the probability of the observed value, conditioned on the parameter, is $e^{-0.5} = 0.607$. Exact solutions are used for $k \leq 5$, and the approximation $k \pm \sqrt{k}$ was used for larger values of k . Selected terms were given equal weights and used as the representation of the query given to the mg system [Witten *et al.*, 1994]. This then produced a “KB”-labeled ranked list of documents for each topic.

3 Compression-Based Relevance Assessment: Scheme *DL*

Our second approach uses the LZW compression method [Welch, 1984] to calculate the entropies of the test documents conditioned on a topic statement plus the documents judged relevant to the topic. These entropies are used to yield probabilities that form the basis for ranking the relevance of new documents for each topic.

In more detail, documents are first preprocessed to remove non-ASCII characters and to replace all whitespace with the space character. SGML heading information is maintained. For each topic the collection of (preprocessed) documents judged relevant to a topic is used to generate a dictionary of strings of various sizes. The topic statement is also included with this collection of documents relevant to the topic, and for ease of exposition, for the remainder of this discussion we will not point this out again — references to collections of documents relevant to a topic should be interpreted to include the topic statement. The dictionaries are formed by initializing them with an alphabet of the printable ASCII characters, plus the space character. The sequence of relevant documents is then parsed greedily into a dictionary: First, the longest substring matching a dictionary entry and starting from the current position (initially, the first character of the document) is found. Then, the substring consisting of the match plus the next character is added as a new dictionary entry. The current position is then moved just after the character following the match, and the process is repeated until end of the sequence of documents has been reached. (See [Ziv and Lempel, 1978] for more details on this dictionary approach for text compression.)

These dictionaries, one per topic, can then be used to compress the test documents (each yielding different degrees of success in compression). Again, the longest substring from the current position (starting at the beginning of the document) matching a dictionary entry is found, and a pointer to the matching entry is output. The current position is moved to just after the match in the document, and the process is repeated until the document has been completely processed. This is the LZW-parse of the document, and compression is achieved when the cumulative size of such pointers is less than the original document length. The entropy estimate for document d conditioned on topic t , written $E(d|t)$, is equal to $p \times \lceil \log_2 |D_t| \rceil$, where $|D_t|$ is the number of entries in the dictionary generated from the documents relevant to topic t , and p is the number of pointers output in the LZW-parse.

Such conditional entropies may be used to calculate conditional probabilities. Let us imagine a machine that, when given a dictionary D_t , repeatedly and randomly selects entries from dictionary D_t and outputs the entries concatenated together until a document of length L is created. Such a machine would generate a particular parse π of document d of length $L = \text{length}(d)$ with probability $P(d, \pi|t) = |D_t|^{-p_\pi}$, where p_π is the number of dictionary entries selected in parse π . The probability of generating any particular document d from D_t is $P(d|t) = \sum_\pi P(d, \pi|t)$.

If we assume that the LZW-parse described is much better than any other parse (*i.e.*, that the inequality $p_{LZW} \ll p_\pi$ holds for $\pi \neq LZW$), then $P(d|t) = kP(d, LZW|t)$, where k is a constant slightly larger than 1. This assumption is known to hold true for natural languages including English [Bell *et al.*, 1990]. Since $P(d, LZW|t) = 2^{-E(d|t)}$, we used a monotonically decreasing function of $E(d|t)$, namely $\rho_{d,t} = \text{length}(d)/E(d|t)$ (where $\text{length}(d)$ is the length of the document) to rank topic relevance for each new document.

4 Scheme-Per-Topic Data Fusion: Scheme *SPT*

Recall that we use the term “scheme” to denote a method for assessing a document’s relevance — computing the similarity (or “proximity”) of either a pair of documents or a document and a query/topic. We now also use the term “scheme” to refer to our two fusion methods, since they, too, are methods for assessing document relevance.

The *SPT* approach simply uses as its relevance-assessment scheme for a topic whichever baseline method works best on that topic. Since we apply data fusion to the *KB* and *DL* schemes, to determine which is better we selected a “training corpus” from the full initial corpus of documents, consisting of the union of 90% of the judged relevant documents, 90% of the judged non-relevant documents, and 10% of the non-judged documents, assumed for training purposes to be not-relevant. This latter collection of documents was selected to broaden the distribution of documents on which the methods would be evaluated. Testing of each scheme took place on the disjoint remainder of the judged and not-judged documents, called the “testing corpus”, with each scheme using just those documents appearing in the training corpus to assess the relevance of documents in the testing corpus.

The sum, over the top 200 documents, of $(200 - \text{rank})$, evaluated at each rank occupied by a relevant document, was then computed, and whichever yielded a better value for a particular topic was used for our official submission on that topic. In this fusion scheme, yielding the results given in our official submission *rutapspt*, the relevance of each document is determined using the complete corpus — the full collection out of which the scheme-selecting training and testing corpora were

selected for our “internal” scheme-selection process.

5 Global Data Fusion: Scheme *GLOB*

For the second fusion scheme we formed a simple global combination rule. Each ranking scheme was used to produce a set of the top 1000 documents. In doing so, each scheme produced a ranking score for each document. (These scores were given by the cosine rule in the *KB* method, and by the compressibility of the document being scored, in the *DL* method.) Scores assigned by each system were normalized, for each topic:

$$\begin{aligned}s_{\max} &= \text{score of top document} \\ s_{\min} &= \text{score of document ranked 1000} \\ s_{\text{norm}} &= (s - s_{\min}) / (s_{\max} - s_{\min}).\end{aligned}$$

If a document did not appear in the top 1000 documents retrieved by scheme *J* it was assigned the normalized score $s_{\text{norm}}^J = 0$. The overall score assigned to a document was then computed as $s_{\text{glob}} = (s_{\text{norm}}^{DL} + s_{\text{norm}}^{KB}) / 2$. Documents in the union of the two lists were ranked in decreasing order of s_{glob} and the top 1000 formed the official submission *rutapglob*.

6 Updated *DL* Results

Nineteen of the routing topics were selected at random. For each topic two dictionaries were formed, one based on the documents that had been judged relevant (called the positive dictionary), and one based on the documents that had been judged not relevant to that topic (call it the negative dictionary). To do this, for each topic half of the relevant, and half of the not relevant documents were selected (once), at random. This represents a single split-half test of the method. The total (9399) of all cases exceed the number of actual documents in the test set (8269) because some documents had been judged not relevant for more than one topic. Each document from the test set for a given topic was compressed with respect to both the positive and negative dictionaries for that topic, and was assigned relevant or not-relevant according to which dictionary gave more compression. The performance is summarized in Table 1, in which the basic data are the numbers of True (False) Positives and Negatives. A True Positive is a relevant document that has been assigned “relevant” by this method.

The results, also presented in Figure 1, are best expressed in terms of the detection rate and the false alarm rate. Together, these define for each topic, a Generalized Retrieval Operating Characteristic (GROC; Kantor and Voorhees, this volume) determined by three points: the indicated operating point, the point (0, 0), and the point (1, 1). For example, the operating point, for this method, for Topic 11 has a false alarm rate of $fa = 52\%$ and a detection rate of $dr = 86\%$. An ideal routing system would have a very low value of fa and a high value of dr .

In the actual TREC situation, the number of non-relevant documents is of the order of 10^6 , while the number of relevant documents is of the order of 10^3 . Thus performance at this level would produce, in general, 860 relevant documents and 520,000 non-relevant documents, for a precision of 0.16%. Clearly if this method is to become part of a workable system some changes are needed.

Topic	#Pos	#Neg	#	TP	TN	FP	FN	FA	DR	Err
001	161	441	602	129	208	233	32	0.53	0.80	0.440
003	308	323	631	303	35	288	5	0.89	0.98	0.464
004	86	374	460	68	297	77	18	0.21	0.79	0.207
005	184	398	582	182	135	263	2	0.66	0.99	0.455
006	169	326	495	158	106	220	11	0.67	0.93	0.467
011	171	359	530	147	174	185	24	0.52	0.86	0.394
012	219	412	631	195	174	238	24	0.58	0.89	0.415
023	113	481	594	108	162	319	5	0.66	0.96	0.545
024	216	281	497	214	60	221	2	0.79	0.99	0.449
044	293	458	751	283	131	327	10	0.71	0.97	0.449
053	348	126	474	346	6	120	2	0.95	0.99	0.257
054	136	122	258	131	58	64	5	0.52	0.96	0.267
058	111	181	292	108	95	86	3	0.48	0.97	0.305
068	131	222	353	123	105	117	8	0.53	0.94	0.354
077	109	202	311	106	122	80	3	0.40	0.97	0.267
078	112	171	283	108	60	111	4	0.65	0.96	0.406
082	363	50	413	362	19	31	1	0.62	1.00	0.077
094	350	250	600	335	67	183	15	0.73	0.96	0.330
095	325	317	642	40	303	14	285	0.04	0.12	0.466
total	3905	5494	9399	3446	2317	3177	459	0.58	0.88	0.387

Table 1: Updated *DL* method. #Pos=Number of positive test examples. #Neg=Number of negative test examples. #=Number of test examples. TP=True (correct) positive classifications. TN=True negative classifications. FP=False positive classifications. FN=False negative classifications. FA=False alarm rate. DR=Detection Rate. Err=Crude Error Rate.

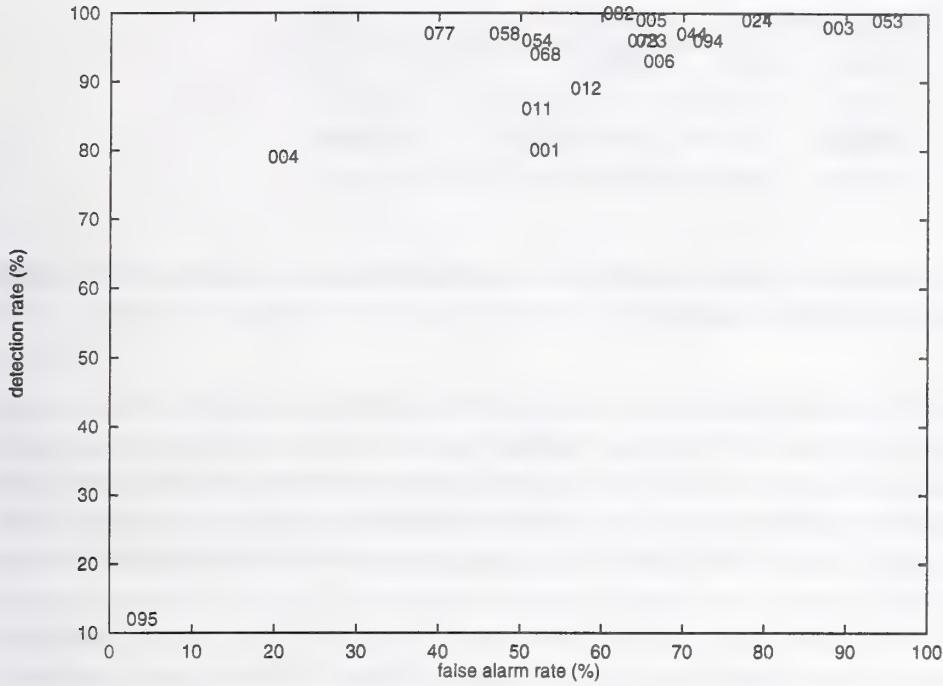


Figure 1: GROC curve for Updated *DL*.

The simplest method to solve this problem would be to convert the compressibilities into conditional probabilities $P(d, LZW|t)$ and calculate *a posteriori* probabilities for the relevant and non-relevant sets:

$$P(t|d, LZW) = \frac{P(d, LZW|t)P(t)}{P(d, LZW)}$$

where $P(d, LZW)$ may be set arbitrarily since it is the same for both the relevant and non-relevant (negative) sets for a given document. Routing would then proceed by classifying as relevant all documents d for which $P(\text{relevant}|d, LZW) > P(\text{non-relevant}|d, LZW)$. Another possible source of improvement is to change the assignment rule, by finding the relative compressions which empirically optimize, for example, precision at 1000 documents retrieved. This can be explored by ranking documents in decreasing order of the ratio of compressions achieved by the positive and negative dictionaries, and choosing the cutoff to optimize a measure of precision. Another possibility is to combine this method with some method which is essentially orthogonal in its principles, to see whether fusion of the two methods gives an improvement over either method.

7 Results

Unfortunately, the implementations of our *DL* and *SPT* schemes both had bugs that were not discovered until after the submission of our official runs, and thus the retrieval performance for both schemes *SPT* and *GLOB* were quite poor. Due to the bug in *SPT* we do not report further on it

win/tie/lose	<i>DL</i>	<i>KB</i>	<i>GLOB</i>	median
<i>DL</i>	—	1/2/42	0/2/43	0/0/45
<i>KB</i>	38/6/1	—	34/10/1	11/7/27
<i>GLOB</i>	35/9/1	4/9/32	—	0/4/41
median	42/3/0	33/10/2	39/5/1	—

Table 2: Performance comparisons: win/draw/lose on number of relevant retrieved in top 1000 (above diagonal) and top 100 (below diagonal).

here. In the case of *GLOB*, which combines the ratings of *KB* and (the incorrectly implemented) *DL*, at 1000 documents *GLOB* equaled the median on four topics, and was inferior on the remaining topics. Results were roughly comparable at 100 documents retrieved, with *GLOB* exceeding the median once, tying it five times, and performing worse in the remaining cases. Nonetheless, these results of our official submissions, as well as our subsequent analyses, allow us to reach a number of conclusions.

First, to calibrate how seriously the implementation error affected the *DL* scheme, we determined what its precision rates would have been for each topic at 100 and 1000 documents retrieved. In every case but one *DL* performed far worse than the median of all official submissions, only equalling median performance for 100 retrieved documents when the median precision was 0. We are therefore confident that our implementation error did not in any way reflect a serendipitous invention of a successful new scheme — it is a vastly inferior method.

In contrast, the *KB* scheme at 1000 retrieved exceeded the median in 11 cases, tying it in 7, and was worse in the remaining cases. At 100 retrieved documents it exceeded the median in 2 cases and tied it in 10. Thus, although it did not perform overwhelmingly well either, it did far better than our incorrectly implemented *DL* scheme.

However, even though incorrectly implemented, the *DL* scheme does represent a scheme that ranks documents, however poorly, and thus we can still consider how well our the *GLOB* data-fusion scheme reconciles the rankings of the *KB* and (incorrect) *DL* schemes. Our ideal result would be to find that *GLOB* performed better than either individual scheme in isolation, demonstrating that fusion can yield better results than either method in isolation. Unfortunately, this was not the case. Occasionally *GLOB* exceeded *KB*, but for the most part *KB*'s performance substantially exceeded that of either fusion method.

On the other hand, the worst-case scenario would be if the fusion method performed worse than the incorrect *DL* scheme. Our results show that this was not the case: *GLOB* exceeded *DL* on every topic but 2, which were ties, at 1000 documents retrieved, and tied in 9 topics, lost in one, and exceeded *DL* on the rest for 100 documents.

These results are summarized in Table 2. Each entry “*a/b/c*” represents the fact that the scheme labeling that row retrieved more relevant documents than the scheme labeling that column on *a* topics, tied it on *b* topics, and retrieved fewer relevant documents on *c* topics. Entries above the diagonal represent data for the number of relevant documents in the top 1000 for each scheme, and entries below the diagonal represent data for the top 100 documents retrieved by each scheme.

However, these results favor cases where few relevant topics were retrieved by any method. In

win/tie/lose	<i>DL</i>	<i>KB</i>	<i>GLOB</i>	median
<i>DL</i>	—	0/0/31	0/0/29	0/0/33
<i>KB</i>	25/0/0	—	34/0/0	11/2/21
<i>GLOB</i>	15/0/0	3/1/21	—	0/0/32
median	26/0/0	25/2/2	28/0/1	—

Table 3: Performance comparisons for cases with more than 10 relevant documents retrieved: win/draw/lose on number of relevant retrieved in top 1000 (above diagonal) and top 100 (below diagonal).

such situations (for example, when the median number of retrieved documents was 0) it is more likely for even a poor method to match the median. To account for this we report the same data in Table 3, but only for those cases where the better of the two schemes being compared in an entry returned more than 10 relevant documents. These results only strengthen the conclusions reached from inspection of Table 2, in most cases skewing data even farther to favor the better of the two schemes.

8 Approximate Term Matching for Retrieval of Degraded Documents

In addition to the routing task, we developed two approaches to the TREC5 confusion track, both based on the idea of approximate matching of substrings in a document. One first approach was based on the *DL* scheme for routing, loosened by allowing approximate matches on dictionary entries in assessing probabilities. Unfortunately we were unable to complete this effort by the submission deadline.

Our second approach, which led to an official submission, was to form 5-grams and base retrieval on how many matches and approximate matches can be found in a document. In more detail, we define a “double-dot-5-gram” to be a 5-gram — a string of five characters — with one position comprising a “don’t care” character that can match from 0 to 2 characters. A double-dot-5-gram matches a string if any substitution of 0, 1, or 2 characters for the don’t care causes the result to be a substring of the string.

A set of double-dot-5-grams was generated for each topic statement by first removing stop words (using the same stopword list used by the SMART system [Salton, 1971], available via ftp from Cornell) and then generating every 5-gram that exists in the resulting document. Thus, for example, given a document containing just the word “orange”, two 5-grams would be generated, “orang” and “range”. Next, each 5-gram is converted into five double-dot-5-grams by one-by-one replacing the character in each position with a don’t care character, generating in our example the double-dot-5-grams “*rang”, “o*ang”, “or*ng”, “ora*g”, and “oran*” from “orang”, if “*” is used to represent a don’t care.

Retrieval was accomplished using a simple match-and-count method. The set of double-dot-5-grams generated from each topic was compared to each line of each corrupted document, and the

number of matches was determined. The documents with the top 1000 cumulative counts across each topic's double-dot-5-grams were returned as the result of the retrieval and were then ordered in decreasing value of the ratio of the number of hits to the number of lines in the document (hits per line).

Although we were pleased to note that for the 5% corrupted output there are 7 cases in which our scores are equal to the best scores and for the 20% corrupted output there are 3 cases in which our scores are equal to the best scores, these were primarily cases in which the median showed that for most methods the document was found amongst the first few returned results. When we compare our results using median ranks, for the 5% corrupted output there are only 3 cases in which we are better than the median, and for the 20% corrupted output there are only 8 cases where we exceed the median.

9 Final Remarks

This paper has described our efforts on the TREC5 routing and confusion tracks. One theme running through these efforts is a search for alternatives to the term-based schemes that presently dominate the field of information retrieval. Whether it is the *KB* scheme's modification of weighting methods to select terms for retrieval, or the more drastic change of perspective embodied in the *DL* method, our attempt has been to expand the variety of document representations with which these problems are approached. We rest firmly on a "statistical" approach to information retrieval, with no overt pretext of addressing semantic issues. However, we do try to exploit other sources of information that could potentially assist in information-access tasks, such as sequences comprising arbitrarily long sets of words that can be considered by, for example, the *DL* scheme. Whether any such automated schemes can achieve the stunning independence of interpretations by distinct humans, which forms the basis for the positive results reported in our earlier data-fusion work [Saracevic and Kantor, 1988, Belkin *et al.*, 1995] remains to be seen.

Acknowledgments

The LZW code used for our *DL* analysis is a modified version of code written by Michiel Noordewier. We thank William Cohen and Daniel Kudenko for their participation in our TREC5 efforts. The support of the Alexandria Project Laboratory for travel expenses is gratefully acknowledged.

References

- [Bartell *et al.*, 1994] B. Bartell, G. Cottrell, and R. Belew. Automatic combination of multiple ranked retrieval systems. *Proceedings of the 17th Annual International AMC SIGIR Conference on Research and Development in Information Retrieval*, 1994.
- [Belkin *et al.*, 1995] N. Belkin, P. Kantor, E. Fox, J. Shaw. Combining the evidence of multiple query representations for information retrieval. *Information Processing and Management*, 31(3)431–448, 1995.
- [Bell *et al.*, 1990] T. Bell, J. Cleary, and I. Witten. *Text Compression*. Prentice Hall, Englewood Cliffs, NJ, USA, 1990.

- [Damashek, 1995] M. Damashek. Gauging similarity via n-grams: Text sorting, categorization and retrieval in any language. *Science* v267 (10 Feb 1995) 843–848.
- [Grossman *et al.*, 1996] D. Grossman, O. Frieder, D. Holmes, M. Nguyen, C. Kingsbury. Improving accuracy and run-time performance for TREC-4. In Harman D (Ed.) Proceedings of TREC-4, 1996.
- [Li and Vitányi, 1993] M. Li and P. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*. Springer-Verlag, New York, 1993.
- [Luhn, 1958] H. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159-165 and 317, April 1958.
- [Ng and Kantor, 1996] K.B. Ng and P. Kantor. Two experiments on retrieval with corrupted data and clean queries in the TREC4 adhoc task environment: Data fusion and pattern scanning. In Harman D (Ed.) Proceedings of TREC-4, 1996.
- [Salton, 1971] G. Salton (ed.) *The SMART Retrieval System – Experiments in Automatic Document Processing*. Prentice Hall, Englewood Cliffs, NJ, 1971.
- [Saracevic and Kantor, 1988] T. Saracevic and P. Kantor. A study of information seeking and retrieving: III Searchers, searches and overlap. *JASIS* 39:178–194, 1988.
- [Turtle and Croft, 1991] H. Turtle and W. Croft. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems*, 9(3)187–222, 1991.
- [Welch, 1984] T. Welch. A technique for high-performance data compression. *IEEE Computer*, 17(6):8–19, 1984.
- [Witten *et al.*, 1994] I. Witten, A. Moffat, and T. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. van Nostrand Reinhold, New York, 1994.
- [Ziv and Lempel, 1978] J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *IEEE Trans. Information Theory*, 24:530–536, 1978.



Report on the TREC-5 Experiment:

Data Fusion and Collection Fusion

Jacques Savoy, Anne Le Calvé, Dana Vrajitoru

Institut interfacultaire d'informatique
Université de Neuchâtel (Switzerland)

Summary

This paper describes and evaluates a retrieval model that considers the problem of data fusion and collection fusion as two faces of the same coin. To establish a clear theoretical foundation for combining various sources of evidence provided either by different search schemes (data fusion) or by distributed information services (collection fusion), we have implemented a retrieval model based on the logistic regression methodology.

Participation: Category B, ad-hoc automatic

Introduction

There exist many reasons for considering multiple sources of evidence in information retrieval (Katzner et al., 1982), (Saracevic & Kantor, 1988), (Harman, 1995), and their integration is usually studied in two distinct contexts. Various retrieval strategies or query formulations may operate on the same collection (data fusion problem) (Belkin et al., 1995), (Lee, 1995), subject described in the first part. The second part deals with the collection fusion problem or how distributed information servers may collaborate to answer to a given request (collection fusion problem) (Callan et al., 1995), (Voorhees et al., 1995).

1. Data Fusion Problem

To combine different retrieval schemes (or different query formulations), a retrieval engine might first find the retrieved set associated with each search scheme, and then merge them into a single effective ranked list. To define this underlying merging function, we may consider, for each retrieved record, its rank and/or its retrieval status value. However, the retrieval status values obtained by various weighting schemes may not have a range of possible similar values, leading to a more complex combination situation.

Section 1.1 outlines our test-collection and some evaluations of individual retrieval schemes based on two distinct query constructions. Section 1.2 presents the main design principles of our logistic search model. The last section depicts an evaluation of various suggested schemes and of our approach to data fusion.

1.1. Evaluation of Individual Retrieval Schemes

Before presenting both fusion problems, an outline of our test-collection and an evaluation of some existing retrieval schemes may be useful. For TREC'5, we have considered the WSJ2 corpus (74,520 documents, and 45 queries) as one collection on the one hand, and on the other, as composed of three distinct sub-collections, according to their respective publication year (see Table 1).

Collection	WSJ90	WSJ91	WSJ92	WSJ2
Size	73 Mb	146 Mb	35 Mb	254 Mb
# of documents	21,705	42,652	10,163	74,520
# of topics	38	40	28	45
# relevant doc.	316	602	146	1,064

Table 1: Collection statistics

The queries (#251 to #300) are fully automatically constructed based on the available natural language description. For each sub-collection, we do not use the same number of topics. More precisely, from the WSJ90 collection, the queries {252, 260, 263, 272, 277, 278, 279, 280, 281, 292, 295, 296} do not have any relevant document and are removed from the evaluation. For the WSJ91 corpus, the "null" topics are {253, 260, 262, 263, 267, 276, 278, 279, 281, 296}, and for the WSJ92 collection, the queries {252, 253, 256, 258, 262, 263, 265, 266, 267, 268, 271, 275, 276, 278, 279, 280, 281, 288, 293, 295, 296, 300} are removed for the same reason. For the whole WSJ2 collection, the queries {263, 278, 279, 281, 296} can be ignored.

The evaluation of various vector-processing schemes and the probabilistic OKAPI model is shown in Table 2 within which the OKAPI performance is used as baseline. For this test, each request was constructed based on either the Descriptive section only or on both the Descriptive and Narrative sections (the precise specifications of these search strategies can be found in Appendix 1).

Assuming that a difference of 5% in average precision can be considered as significant, we may conclude that the Narrative section contains important search terms. Thus, the inclusion of this logical section may significantly improve the retrieval effectiveness. As an exception to this rule, the HTN-BNN

scheme seems to perform better with short queries than with long requests. From this data, we may also conclude that the OKAPI, LNU-LTC and ATN-NTC search models result in significant enhancement over other vector-processing schemes. Moreover, simple weighting schemes which are based on binary indexing (BNN-BNN) or only on term frequencies (NNN-NNN) must be clearly discarded.

Collection Model	Precision (% change)	
	WSJ2 <desc> 45 queries	WSJ2 <desc>&<narr> 45 queries
Individual Retrieval Scheme		
OKAPI - NPN (baseline)	14.06	20.30
LNU - LTC	15.00 (+6.76)	20.47 (+0.84)
ATN - NTC	14.54 (+3.49)	20.48 (+0.89)
LTN - NTC	13.48 (-4.06)	18.58 (-8.47)
LNC - LTC	11.54 (-17.86)	18.51 (-8.82)
LTC - LTC	10.07 (-28.32)	14.65 (-27.83)
ANN - NTC	12.91 (-8.11)	17.04 (-16.06)
ANC - LTC	8.10 (-42.35)	16.39 (-19.26)
HTN - BNN	15.17 (+7.97)	13.35 (-34.24)
LNC - LNC	5.95 (-57.65)	13.20 (-34.98)
ANN - ANN	10.05 (-28.47)	7.80 (-61.58)
NNN - NNN	2.52 (-82.06)	3.68 (-81.87)
BNN - BNN	4.57 (-67.47)	3.45 (-83.00)

Table 2: Evaluation of Individual Retrieval Schemes

1.2. Our Logistic Retrieval Model

From previous research projects, we may conclude that the combination of various retrieval schemes represents a useful strategy for enhancing the retrieval effectiveness, specially when combining multiple queries formulations (Turtle & Croft, 1991), (Shaw & Fox, 1994), (Belkin et al., 1995).

From our point of view, we consider that formulating a request is a difficult task for the users, and asking them to specify two or more queries may render this process more complex. Therefore, we think it is more appropriate to work with a single request formulation. Moreover, most of the previous works make use of heuristics to merge the results of separate search strategies, and only take account of the retrieval status value as an explanatory variable.

To overcome these difficulties, we suggest using the logistic regression (Hosmer & Lemeshow, 1989) as a methodology for combining multiple sources of evidence regarding the relevance of a given document. Of course, this statistical approach has been already applied in other domains such as informetrics (Bookstein et al., 1992) or as a retrieval model (Gey, 1994), (Fuhr & Pfeifer, 1994).

In our approach, we may estimate the probability of a given document D_i 's relevance by computing the following formula:

$$\text{Prob}[D_i \text{ is relevant} \mid x] = \pi(x) = \frac{1}{1 + e^{-\alpha - \beta \cdot x}} \quad (1)$$

within which $\beta \cdot x = \sum_{j=1}^r \beta_{1j} \cdot \text{RANK}_j(D_i) + \beta_{2j} \cdot \text{RSV}'_j(D_i) + \beta_{3j} \cdot \text{VARIA}_j(D_i)$

As shown in Equation 1, our model may take account of the rank, the retrieval status value (without any normalization) and the variation (VARIA) of the retrieval status value compared to the highest value that can be achieved by the corresponding request and search model. For example, within the coordination match model (BNN-BNN), this highest value is defined as the number of search terms.

Model	β_{1j} (RANK)	β_{2j} (RSV')	β_{3j} (VARIA)
OKAPI - NPN	not signif.	0.049	not signif.
LNU - LTC	not signif.	not signif.	0.0255
LTN - NTC	-0.00045	0.141	not signif.
LNC - LTC	-0.00033	not signif.	0.0342
LTC - LTC	not signif.	not signif.	0.0197
LNC - LNC	-0.00015	not signif.	not signif.
ATN - NTC	-0.00022	0.195	not signif.
constant α	-6.0871		

Table 3a: Logistic Regression Coefficient Values (Topic = <desc>)

Based on the WSJ2 corpus and 147 queries (#51 to #250), we have computed the corresponding coefficient values using the SAS package. From the resulting data depicted in Table 3b, we may reach the conclusion that the retrieval schemes LNU-LTC, LTC-LTC, and LNC-LNC do not have a real impact in our logistic model. These search strategies can thus be ignored (the label "not signif." means that the value of the coefficient can be statistically considered as 0). This conclusion however does not mean that these search models are without any

merit, but rather that their influence is already taken into account by the remaining search strategies. For these models, we can also conclude that the rank and either the RSV or the variation are statistically good predictors.

Model	β_{1j} (RANK)	β_{2j} (RSV')	β_{3j} (VARIA)
OKAPI - NPN	-0.00056	not signif.	0.0923
LNU - LTC	not signif.	not signif.	not signif.
LTN - NTC	-0.0004	not signif.	0.00981
LNC - LTC	-0.00051	6.17	not signif.
LTC - LTC	not signif.	not signif.	not signif.
LNC - LNC	not signif.	not signif.	not signif.
ATN - NTC	-0.00064	0.448	not signif.
constant α	-5.8734		

Table 3b: Logistic Regression Coefficient Values (Topic = <desc> & <narr>)

1.3. Evaluation

In evaluating our logistic model, we are also interested to compare its performance with both individual schemes (see Table2), and with other data fusion strategies. To address this second point, we have implemented a data fusion model derived from the studies of Fox & Shaw (1994) and Lee (1995). After selecting the same individual retrieval strategies (given in Table4), we first divide the retrieval status value by the maximum of those achieved in the corresponding list (see Equation2).

$$RSV(D_i) = RSV'(D_i) / \text{Max} \{RSV'(D_i)\} \quad (2)$$

in which $RSV'(D_i)$ indicates the retrieval status value obtained by document D_i .

Second, given a set of r retrieval schemes, each producing a normalized retrieval status value $RSV_j(D_i)$, we may combine these multiple sources of evidence according to the following formula:

$$RSV(D_i) = \bigoplus_{j=1}^r \alpha_j \cdot RSV_j(D_i) \quad (3)$$

within which the parameters α_j indicate the relative weight associated with each retrieval scheme, and \oplus the operator to be applied to combine the retrieval status values. The addition seems to be the best operator \oplus (Belkin et al., 1995) and was selected during our evaluation. Moreover, previous reports indicate that an appropriate value for the parameters α_k seems to be a constant (e.g., 1 as defined in our Model1 in Table4). However, we may weight the relative importance of

each retrieval scheme based on their relative retrieval performance (see Table2) leading to the definition of our Model2 in Table4.

Model	Model 1 α_k	Model 2 α_k
doc. = OKAPI, query = NPN	1	2
doc. = LNU, query = LTC	1	2
doc. = LTN, query = NTC	1	1.5
doc. = LNC, query = LTC	1	1.5
doc. = LTC, query = LTC	1	1
doc. = LNC, query = LNC	1	1
doc. = ATN, query = NTC	1	1.5

Table 4: Parameters Specification

Collection Model	Precision (% change)	
	WSJ2 <desc> 45 queries	WSJ2 <desc>&<narr> 45 queries
Best individual scheme (HTN / ATN)	15.17	20.48
Data fusion Model 1, \oplus = SUM	15.15 (-0.13)	22.45 (+9.62)
Data fusion Model 2, \oplus = SUM	14.98 (-1.25)	22.58 (+10.25)
Logistic regression RANK _k (D _i), RSV' _k (D _i), VARIA _k (D _i)	15.97 (+5.27)	22.72 (+10.94)
Official names	UniNE7	UniNE8

Table 5: Evaluation of Data Fusion Strategies

Compared to the best individual run, our data fusion model presents a significant enhancement. Moreover, the query length seems to play an important role in data fusion Model1 and2, leading to the conclusion that such data fusion approach may further improve only long queries.

2. Collection Fusion

After selecting the more appropriate sources of information (collection selection problem), a collection fusion strategy must provide a mean of effectively merging multiple independent retrieval results into a single ranked list. Section 2.1 describes related research for resolving the collection fusion problem. Our suggested logistic model is presented in Section 2.2, while the last section depicts an evaluation of some of these strategies.

2.1. Related Works on Collection Fusion

Recent works in this domain have suggested some solutions to the merging of separate answer lists obtained from distributed information services. As a first approach, we might assume that (1) the answer lists obtained from various information servers contain only the ranking of the retrieved items, and (2) that each sub-collection contains roughly the same number of relevant items for each submitted request. In such circumstances, we may interleave the results in a round-robin fashion.

As a second method, we might formulate the hypothesis that each information server applies the same (or very similar) search strategy and that the document score values are directly comparable. Such a strategy, called raw-score merging, produces a final list based on the retrieval status value computed by each sub-collection. However, as demonstrated by Dumais (1995), collection-dependent statistics in document or query weights may vary widely among sub-collections, and therefore, this phenomenon may invalidate the raw-score merging hypothesis.

Finally, Callan et al. (1995) suggest a merging strategy based on the score achieved by both sub-collection and document. Therefore, in this scheme, the sub-collections are ranked according to the probability that they respond appropriately to the current request. This strategy produces a performance similar to a run treating the entire set of documents as a single collection.

2.2. Our Logistic Retrieval Model

In our model, we have analyzed and designed a logistic regression for each information server or sub-collection participating in the final result. To determine the relevance probability for a given document D_i in a given sub-collection, we propose computing the following value:

$$\text{Prob}[D_i \text{ is relevant} \mid \mathbf{x}] = \pi(\mathbf{x}) = \frac{1}{1 + e^{-\beta \cdot \mathbf{x}}} \quad (4)$$

with $\beta \cdot \mathbf{x} = \beta_0 + \beta_1 \cdot \text{RANK}(D_i) + \beta_2 \cdot \text{RSV}'(D_i) + \beta_3 \cdot \text{VARIA}(D_i)$

In this case, the different values $\pi(\mathbf{x})$ may be compared directly with the various sub-collections or search strategies. For the merging procedure, these estimated relevance probabilities define the sort key (or the number and the final position) for each item extracted from each sub-collection. In its actual form, our

retrieval scheme does not include a sub-collection selection procedure that, based on the current request, may automatically pick out sub-collections forming part of the final solution. Thus, each query is submitted to all sub-collections and the resulting lists are merged according to the different values of $\pi(x)$.

Model	β_0 (CONSTANT)	β_1 (RANK)	β_2 (RSV')	β_3 (VARIA)
OKAPI - NPN	-5.9763	-0.00317	0.093	0.0839
LNU - LTC	-5.2181	-0.00343	210.9	0.0178
LNC - LTC	-5.9942	-0.00451	22.2	not signif.

Table 6a: Logistic Regression Coefficient Values (Topic = <desc>)

Model	β_0 (CONSTANT)	β_1 (RANK)	β_2 (RSV')	β_3 (VARIA)
OKAPI - NPN	-5.1710	-0.00515	0.028	0.0853
LNU - LTC	-5.1253	-0.00402	220.8	0.0174
LNC - LTC	-5.7964	-0.00668	22.3	0.0

Table 6b: Logistic Regression Coefficient Values (Topic = <desc> & <narr>)

Based on the data in Table6, one can see that the coefficient values of our logistic model are very similar when comparing the short and long queries.

2.3. Evaluation

As described in Tables7, our collection fusion problem is particular. As usual, we have divided a test-collection into various sub-collections. However, in this paper, we apply different retrieval schemes to each sub-collection.

Under such circumstances, the round-robin strategy presents relatively interesting performance, while the raw-score merging strategy is clearly ineffective. In fact, all the retrieved documents are extracted from the WSJ90 collection because the OKAPI model retrieval status values are always greater than those of the LNU-LTC or LNC-LTC search schemes (see statistics given in Tables7). If we normalize the retrieval status value within each sub-collection by dividing them by the maximum RSV of each result list, the retrieval performance is always significantly worse than the round-robin strategy.

Our logistic model presents a significant enhancement over the round-robin scheme when dealing with short queries (Table7a) and similar performance with long requests (Table7b).

Collection Model	Precision (% change)		
	WSJ90 OKAPI - NPN	WSJ91 LNU - LTC	WSJ92 LNC - LTC
Average precision	38 queries 17.22	40 queries 16.29	28 queries 17.52
# of relevant doc.	316	602	146
# of relevant doc. retrieved	229	400	127
RSV min	2.037	0.002	0.011
RSV max	34.136	0.023	0.309
RSV mean	5.513	0.006	0.0518
RSV standard error	2.433	0.00236	0.0253
Collection Fusion	WSJ2		
# of relevant doc.	45 queries 1064		
Round-robin (baseline)	11.38		
Raw-Score Merging	5.89 (-48.24)		
Norm. Raw-Score Merging	9.17 (-19.42)		
Logistic RANK(D_i), RSV(D_i), VARIA(D_i)	13.35 (+17.31)		
Official name	UniNE0		

Table 7a: Evaluation of Collection Fusion Strategies (Topic = <desc>)

Collection Model	Precision (% change)		
	WSJ90 OKAPI - NPN	WSJ91 LNU - LTC	WSJ92 LNC - LTC
Average precision	38 queries 28.86	40 queries 19.48	28 queries 21.45
# of relevant doc.	316	602	146
# of relevant doc. retrieved	249	430	130
RSV min	4.588	0.002	0.017
RSV max	107.889	0.025	0.288
RSV mean	17.937	0.0067	0.05737
RSV standard error	8.926	0.002	0.023
Collection Fusion	WSJ2		
# of relevant doc.	45 queries 1064		
Round-robin (baseline)	19.75		
Raw-Score Merging	12.64 (-36.00)		
Norm. Raw-Score Merging	13.18 (-33.27)		
Logistic RANK(D_i), RSV(D_i), VARIA(D_i)	19.85 (+0.51)		
Official name	UniNE9		

Table 7b: Evaluation of Collection Fusion Strategies (Topic = <desc> & <narr>)

Conclusions and Future Work

This paper describes a unified approach to combining multiple sources of evidence to both the problems of data fusion and collection fusion. To resolve these two distinct questions, we have used the same design principles, algorithm and data structures, showing that the resulting logistic model reveals particularly interesting retrieval effectiveness. Moreover, the evaluation results depicted in this paper demonstrated that the Narrative section of TREC topics has a clear and positive impact on the retrieval effectiveness.

In the near future, we will address the following questions:

- a) Data fusion problem: based only on one query formulation, our experience seems to indicate that it is important to consider only two or three retrieval schemes instead of six. Is it always the case and why?
- b) Collection fusion problem: when only the rank is available as explanatory variable, how can we use our logistic approach, and does such a retrieval model present a significant enhancement over the round-robin strategy?
- c) Are the values of the logistic regression coefficients obtained with one tested collection (WSJ2) valid for another corpus (e.g., SJMN)?

Finally, in this study, we never take known relevant documents (Salton & Buckley, 1990) or pseudo-relevance information into account (Buckley et al., 1995) in order to improve retrieval effectiveness. Although we do not reject this attractive proposition, our objective is to evaluate the effectiveness of the initial search. Relevance feedback can therefore be used after this first search in order to enhance the retrieval performance.

Acknowledgments

The authors would like to thank C. Buckley for giving us the opportunity to use the SMART system, without which this study could not have been conducted. This research was supported by the SNSF (Swiss National Science Foundation) under grant 20-43'217.95.

References

- Belkin, N. J., Kantor, P., Fox, E. A. & Shaw, J. A. (1995). Combining the evidence of multiple query representations for information retrieval. *Information Processing & Management*, 31(3), 431-448.
- Bookstein, A., O'Neil, E., Dillon, M. & Stephens, D. (1992). Applications of loglinear models for informetric phenomena. *Information Processing & Management*, 28(1), 75-88.
- Buckley, C., Singhal, A., Mitra, M. & Salton, G. (1995, November). *New retrieval approaches using SMART*. Proceedings of the TREC'4, Gaithersburg, MD, in press.
- Callan, J. P., Lu, Z. & Croft, W. B. (1995, June). *Searching distributed collections with inference networks*. Proceedings of the ACM-SIGIR'95, Seattle, WA, 21-28.
- Dumais, S. T. (1993, November). *Latent semantic indexing (LSI) and TREC-2*. Proceedings of TREC'2, Gaithersburg, MD, NIST Publication #500-215, 105-115.
- Fuhr, N. & Pfeifer, U. (1994). Probabilistic information retrieval as a combination of abstraction, inductive learning, and probabilistic assumptions. *ACM Transactions on Information Systems*, 12(1), 92-115.
- Gey, F. C. (1994, July). *Inferring probability of relevance using the method of logistic regression*. Proceedings of the ACM-SIGIR'94, Dublin, Ireland, 222-231.
- Harman, D. (1995). Overview of the second text retrieval conference (TREC-2). *Information Processing & Management*, 31(3), 271-289.
- Harman, D. (1986). *An experimental study of factors important in document ranking*. Proceedings of the ACM-SIGIR'86, Pisa, Italy.
- Hosmer, D. & Lemeshow, S. (1989). *Applied logistic regression*. New-York, NY: John Wiley & Sons.

- Katzer, J., McGill, M. J., Tessier, J. A., Frakes, W. & DasGupta, P. (1982). A study of the overlap among document representations. *Information Technology: Research & Development*, 2, 261-274.
- Lee, J. H. (1995, July). *Combining multiple evidence from different properties of weighting schemes*. Proceedings of the ACM-SIGIR'95, Seattle, WA, 180-188.
- Salton, G. & Buckley, C. (1990). Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4), 288-297.
- Saracevic, T. & Kantor, P. (1988). A study of information seeking and retrieving. III. Searchers, searches, overlap. *Journal of the American Society for Information Science*, 39(3), 197-216.
- Shaw, J. A. & Fox, E. A. (1994, November). *Combination of multiple searches*. Proceedings of the TREC'3, Gaithersburg, MD, NIST Publication #500-225, 105-108.
- Turtle, H. & Croft, W. B. (1991). Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems*, 9(3), 187-222.
- Voorhees, E. M., Gupta, N. K. & Johnson-Laird, B. (1995, July). *Learning collection fusion strategies*. Proceedings of the ACM-SIGIR'95, Seattle, WA, 172-179.

Appendix 1: Weighting Schemes

In this paper, the indexing procedure done by the SMART system is fully automatic and based on a single term only. The representation of each topic is based on the content of its Descriptive (<desc>) section or its Descriptive and Narrative (<narr>) sections. For each document, the Text (<text>) section as well as the Subtitle (<ST>), Headline (<HL>), and Summary (<LP>) sections were used to build the document surrogate. All other subsections were removed, and, in particular, the title and the concept section of each topic (see TableA.1).

Collection	Section
WSJ2	<desc>, <text>, <st>, <hl>, <lp>
Query	<desc> or <desc> & <narr>

Table A.1: Selected Sections Used to Represent Documents and Queries

To assign an indexing weight w_{ij} reflecting the importance of each single-term T_j , $j=1,2,\dots,t$, in a document D_i , we may use one of the equations shown in Table A.2. In this table, tf_{ij} depicts the frequency of the term T_j in the document D_i (or in the request), n represents the number of documents D_i in the collection, df_j the number of documents in which T_j occurs, and idf_j the inverse document frequency ($\log [n/df_j]$). Moreover, the document length of D_i (the number of indexing terms) is noted by nt_i , and $\text{mean}(nt.)$ indicates the collection mean. The constant c is fixed to 0.2 and C is computed as $0.5+1.5\cdot[nt_i/\text{mean}(nt.)]$. Finally, the computation of the retrieval status value is based on the inner product.

BNN	$w_{ij} = 1$	NNN	$w_{ij} = tf_{ij}$
ANN	$w_{ij} = 0.5 + 0.5 \cdot \frac{tf_{ij}}{\max tf_{i.}}$	ATN	$w_{ij} = \left[0.5 + 0.5 \cdot \frac{tf_{ij}}{\max tf_{i.}} \right] \cdot idf_j$
NPN	$w_{ij} = tf_{ij} \cdot \log \left[\frac{n-df_j}{df_j} \right]$	LTN	$w_{ij} = [\log(tf_{ij})+1] \cdot idf_j$
HTN	$w_{ij} = \frac{\log(tf_{ij}+1) \cdot idf_j}{\log(nt_i)}$	OKAPI	$w_{ij} = \sum_{k=1}^t \frac{2 \cdot tf_{ik}}{C + tf_{ik}}$
LNC	$w_{ij} = \frac{\log(tf_{ij})+1}{\sqrt{\sum_{k=1}^t (\log(tf_{ik})+1)^2}}$	NTC	$w_{ij} = \frac{tf_{ij} \cdot idf_j}{\sqrt{\sum_{k=1}^t (tf_{ik} \cdot idf_k)^2}}$

ANC	$w_{ij} = \frac{0.5 + 0.5 \cdot \frac{tf_{ij}}{\max tf_{i.}}}{\sqrt{\sum_{k=1}^t \left[0.5 + 0.5 \cdot \frac{tf_{ik}}{\max tf_{i.}} \right]^2}}$
LTC	$w_{ij} = \frac{[\log(tf_{ij})+1] \cdot idf_j}{\sqrt{\sum_{k=1}^t ([\log(tf_{ik})+1] \cdot idf_k)^2}}$
LNU	$w_{ij} = \frac{\frac{1 + \log(tf_{ij})}{1 + \log(\text{mean}(tf_{i.}))}}{(1-c) \cdot \text{mean}(nt_{.}) + c \cdot nt_i}$

Table A.2: Weighting Schemes

Using Relevance to Train a Linear Mixture of Experts

Christopher C. Vogt, Garrison W. Cottrell,
Richard K. Belew, Brian T. Bartell
University of California, San Diego
Computer Science and Engineering 0114
La Jolla, CA 92093
{vogt,gary,rik}@cs.ucsd.edu
bbartell@verity.com

Abstract

A linear mixture of experts is used to combine three standard IR systems. The parameters for the mixture are determined automatically through training on document relevance assessments via optimization of a rank-order statistic which is empirically correlated with average precision. The mixture improves performance in some cases and degrades it in others, with the degradations possibly due to training techniques, model strength, and poor performance of the individual experts.

1 INTRODUCTION

The mixture of experts approach is one which is gaining in popularity in many areas of computer science and artificial intelligence (e.g., [Jordan and Jacobs, 1994]) and one which is especially applicable to information retrieval, since in practice the sets of relevant documents returned by different IR algorithms (or *experts*) often have little overlap. In fact, the pooling method used by past TREC's to determine which documents are relevant can be viewed as a sort of mixture model on the grandest of scales [Harman, 1995b]. Each organization represents an expert, and only the top-rated documents from each are passed along to the human judges. The most general mixture model is extremely flexible since it can incorporate any number of

experts. Thus it can subsume any other approach by merely including it as another expert.

The two main difficulties with mixture models are first determining the class of models to use and then finding those model parameters which maximize the performance of the system. Bartell, Cottrell, and Belew have explored both of these issues in some depth - focusing primarily on both linear and nonlinear neural net models coupled with the use of optimization of rank-order statistics to determine model parameters [Bartell et al., 1994b, Bartell et al., 1994a]. Even with the simplest linear combination of experts, they achieved some impressive improvements - up to 47% higher average precision than the best individual expert. All of their experiments, however, are on relatively small collections. Others have successfully used mixture (a.k.a. fusion) approaches on larger collections, including TREC, but they hand pick the model parameters, clearly an undesirable approach (see [Kantor, 1995], [Knaus et al., 1995], [Shaw and Fox, 1995] in [Harman, 1995a]). Here we show how the mixture technique coupled with automatic parameter adjustment via rank-order statistic optimization scales up to the TREC collection. Our results indicate that we have yet to find the best class of model for this task.

2 METHOD

2.1 The TREC Tasks

The Text REtrieval Conference (TREC) has two main tracks: *ad hoc* and *routing*. Furthermore, participants may choose to take part in one of three categories, category A (all of the data) category B (a subset of the data), or category C (for companies who wish only to submit results, and not a paper). We participated in both tracks, category B. In both tasks, participants are given a training set of documents and queries, along with relevance assessments. For the *ad hoc* task, a new set of queries are then distributed and the task is to find those training documents which satisfy the new queries. The *routing* task addresses the converse problem: a subset of the training queries are selected and the goal is to find relevant documents from a new collection. For category B, the training set consisted of 74,520 Wall Street Journal articles (about 253Mb of data) with an average of about 300 terms per document, and 250 queries of varying lengths (from 8 to 180 terms, average of about 20). The *ad hoc* task added 50 new queries with average length of 83 terms (16 for the short version). The average number of relevant documents per *ad hoc* query was about 24, although some queries had no relevant documents. For the *routing* task, 61,578 Foreign Broadcast Information Service documents were used (about 225Mb). For the 50 training queries selected from the training set as *routing* queries, the average length was 83 terms, and the average number of relevant documents was 63.

2.2 The Experts

Our approach was to use three experts, two based on the standard Vector Space model [Salton and Buckley, 1988] and one based on Latent Semantic Indexing [Deerwester et al., 1990]. The two VS models were implemented via Cornell's SMART system [Salton, 1971]. Two very different weighting schemes were used:

one was binary and the other was logarithmically scaled tf-idf. The SMART codes for these two weighting schemes are “bnn” and “ltc” respectively, and we will refer to them using those codes throughout this paper. Since we participated as category B, only those 74,520 Wall Street Journal articles on the second TIPSTER disk were indexed for training and testing the adhoc task, and only the 61,578 FBIS articles were indexed for testing the routing task. Within these documents, only those sections delimited by <TEXT>, <LP>, <HL>, and <IN> were indexed, and were stopped and stemmed using the default SMART routines (with no special treatment for proper nouns). This produced document vectors with 104,113 (WSJ/adhoc) and 188,142 (FBIS/routing) components.

Queries were processed in a similar but slightly more complicated manner. Each topic was first lexically analyzed, and those phrases (as delimited by ‘,’;’, ‘unless’, and ‘except’) which contained the string “not” followed by “relevant,” “about,” or “sufficient” were removed. All other non-SGML text was used, except those terms not found in any training document. The resulting test was indexed using SMART and both bnn and ltc weightings. Furthermore, the adhoc queries were indexed twice, once using the all of the parsed text of the topics, and once using only the <DESC> field (the so-called “short topics”).

The LSI expert mimicked the approach used by Dumais in TREC-3 [Dumais, 1995]. Specifically, since doing a Singular Value Decomposition on the full 104,113 × 74,520 term-by-document matrix from the ltc expert would have taken much too long, it was subsampled. Only those terms occurring in 5 or more documents were used, and only a randomly selected subset of about 10% of the documents was used, leaving a 26,395 × 7500 matrix which was reduced down to 300 × 7500 via SVD using SVDPACKC [Berry, 1992]. This produced a 300 dimensional representation for all the WSJ documents, and the corresponding representations for the queries and FBIS documents were obtained by first removing any terms that were not in the reduced WSJ vocabulary, and then projecting the resulting 26,395-vector down to 300 dimensions as described in [Dumais, 1995].

Once document and query vectors were fixed, relevance scores were computed using the standard inner product rule.

2.3 Combining Scores

Bartell showed that one very effective measure of how well an IR system performs is one which compares the rank-ordering produced by the system to that specified by relevance feedback [Bartell, 1994]. This is in contrast to one which attempts to reproduce the user’s relevance scores exactly. Specifically, Bartell defines a criterion (hereafter called *J*) based on Guttman’s Point Alienation statistic as follows.

Defn: the ranking function implemented by an IR system is

$$R : \Theta \times D \times Q \rightarrow \Re$$

where

Θ = the set of system parameters
 D = the set of document vectors
 Q = the set of query vectors

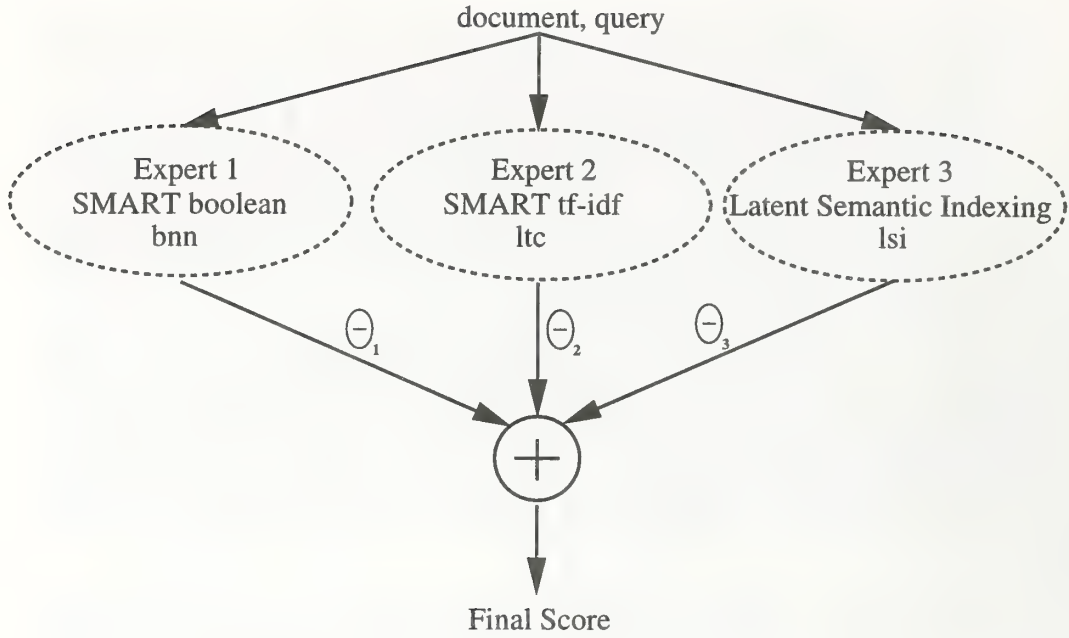


Figure 1: Linear Combination of Relevance Scores

Defn: Bartell's J criterion is:

$$J(R_\theta) = \frac{1}{|Q|} \sum_{q \in Q} \frac{\sum_{d \succ_q d'} (R(\theta, d, q) - R(\theta, d', q))}{\sum_{d \succ_q d'} |R(\theta, d, q) - R(\theta, d', q)|}$$

where $d \succ_q d'$ indicates document d is preferred to document d' on query q .

Note that J has a maximum value of 1 when the numerator and denominator are the same (i.e., the IR system ranks documents exactly as the user would), and a minimum value of -1 when the opposite is true.

Since Bartell rarely did better by using nonlinear models, and because such models tend to have a much larger set of parameters Θ , we chose to use a very simple linear mixture model, namely:

$$R_{mix} = \theta_{bnn} R_{bnn} + \theta_{ltc} R_{ltc} + \theta_{lsi} R_{lsi} \quad (1)$$

which has only three parameters. This is illustrated in Figure 1. Furthermore, other work by Bartell [Bartell, 1994] showed that if only the 15 top-ranked documents for each query (as determined by one of the experts) are used when optimizing J , *better* results are actually achieved. Thus, when we trained our model, we initially used only the top 15 documents (according to the ltc expert). The J criterion was maximized using a simple hill-climbing algorithm with multi-start. Starting from 5 different points in Θ space always yielded a $J \approx 0.59$, and more importantly, the Θ vectors at the maxima were all multiples of each other. Thus, with such a simple model, the J surface appears to have a set of global maxima, and the simple-minded hill-climbing approach to optimization is reasonable.

Expert	Mean (StdDev) for R_{expert}	θ_{expert}	Mean $R \times \theta$	% of total
bnn	10.6 (8.2)	1.0000	10.6	46
ltc	0.21 (0.076)	24.855	5.14	23
lsi	0.68 (0.095)	10.377	7.04	31

Table 1: Parameters and Relative Weightings for Training on the Top 15 Documents

Expert	Mean (StdDev) for R_{expert}	θ_{expert}	Mean $R \times \theta$	% of total
bnn	9.7 (7.4)	0.049	0.48	26
ltc	0.15 (0.062)	2.769	0.43	23
lsi	0.65 (0.086)	1.431	0.93	51

Table 2: Parameters and Relative Weightings for Training on the Top 100 Documents

Unfortunately, training on the top 15 documents for each query led to counter-intuitive results: the bnn expert was weighted more heavily than the other two. This can be seen in Table 1, where the bnn expert gets twice the weight of the ltc expert and 50% more than the lsi expert. This does not seem reasonable since the variance in the scores of the bnn expert is very high, suggesting it may be an unreliable judge of relevance.

Since Bartell’s experiments with the top 15 documents were performed on collections an order of magnitude or smaller than the WSJ collection, we increased the number of documents used for training to the top 100 for each query. Once again, multi-start optimization always led to the same maximum of $J \approx 0.66$ and parameter vectors at the maxima were multiples of each other. These results, summarized in Table 2, show a more reasonable combination, with the lsi expert counting for half of the overall ranking score and the other two splitting the remainder nearly equally.

The three θ_{expert} values from Table 2 were used to combine the relevance scores of the three experts according to equation 1. The top 1000 ranked documents were submitted for three runs: one in which the routing queries were run against the FBIS documents, one where the full adhoc queries were run against the WSJ documents, and a third which ran the short-topic versions of the adhoc queries against WSJ.

3 RESULTS

Precision/Recall curves for each expert and the mixture determined by training on the top 100 documents are shown below for the three runs. Also included are the performance curves for the individual experts on those documents used for training (top 100) and those available for training, but which were not used to select Θ . The latter run is included to explore how well our technique of training on the top n

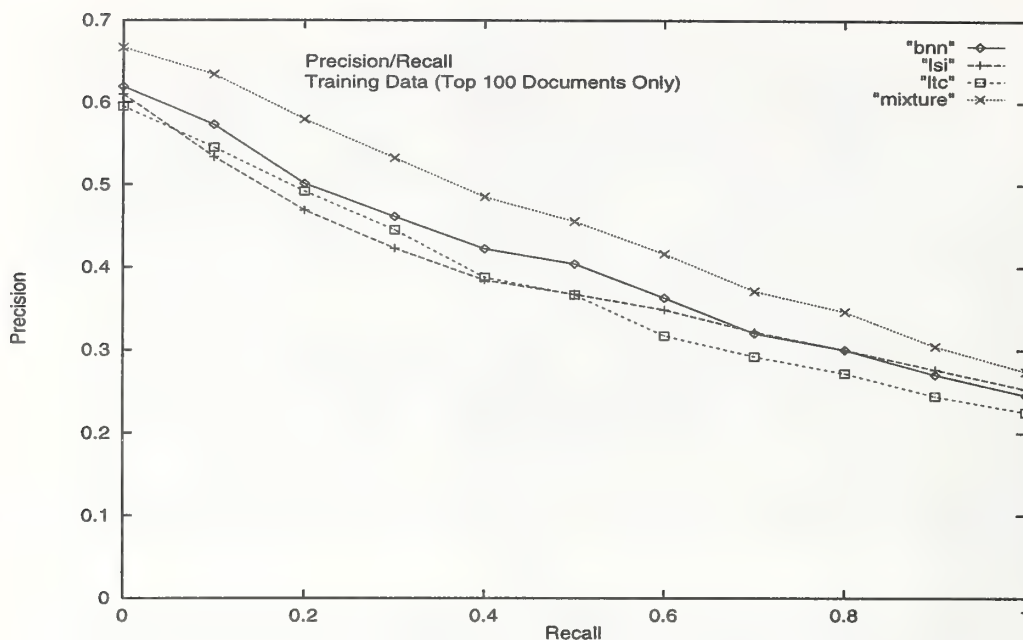


Figure 2: Performance on the Actual Training Data (Top 100 Documents)

documents generalizes to the entire available training set.

A couple of predictable trends are evident in the results. First, the short-topic version of the adhoc task performs worse than the version which uses all topic text. Second, as has been the case in past TREC's, the routing results are generally better than the adhoc ones.

Surprisingly, the lsi expert does not generally outperform the ltc expert on any of the runs. This is in contrast to what Dumais, et al. have found in past TREC's. However, this is not critical for the ideas we are exploring with this paper, since in theory any set of experts may be used regardless of their performance.

The interesting part of these results comes in comparing the mixture's performance with those of the individual experts. Ironically, this shows mixed results. As expected, the mixture performs significantly better than any individual expert on the actual training data (top 100 documents) (see Figure 2). This is expected because previous experiments have suggested that J is highly correlated with average precision, so by optimizing J , we are also optimizing precision. However, when the same mixture is applied to the entire available training set, it only does as well as the second-best expert (lsi) (Figure 3). Thus, generalization is poor to the entire training set, indicating that we may have overtrained, that our model is not powerful enough, or that the top 100 documents do not accurately represent the entire data set. The first explanation seems the least plausible, since there are only 3 parameters in our model. The second two are ideas we will have to explore in future TREC's. On a more positive note, for both versions of the adhoc task,

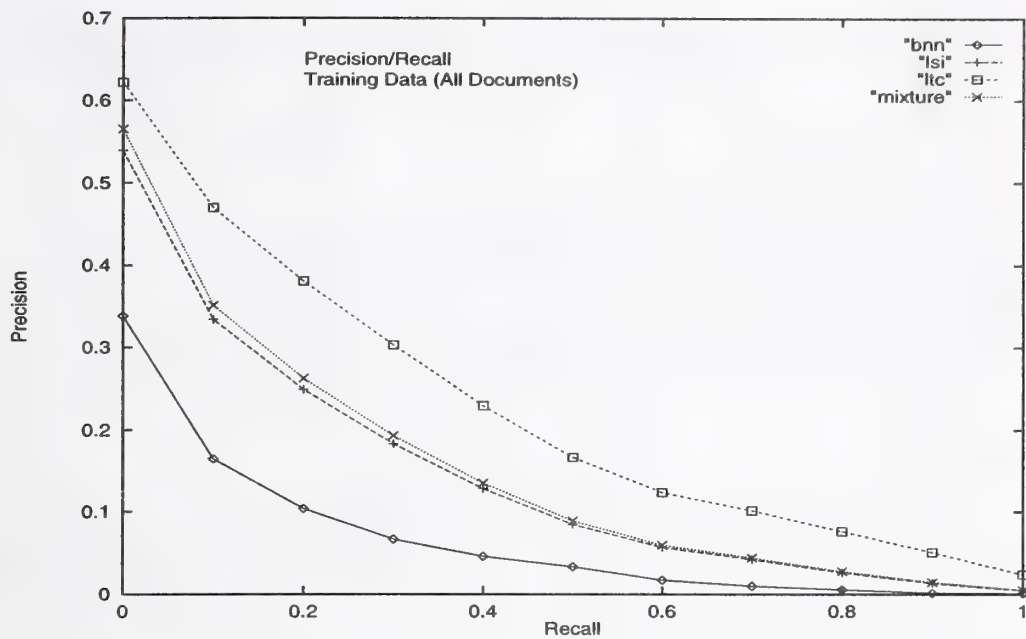


Figure 3: Performance on the Available Training Data (All Documents)

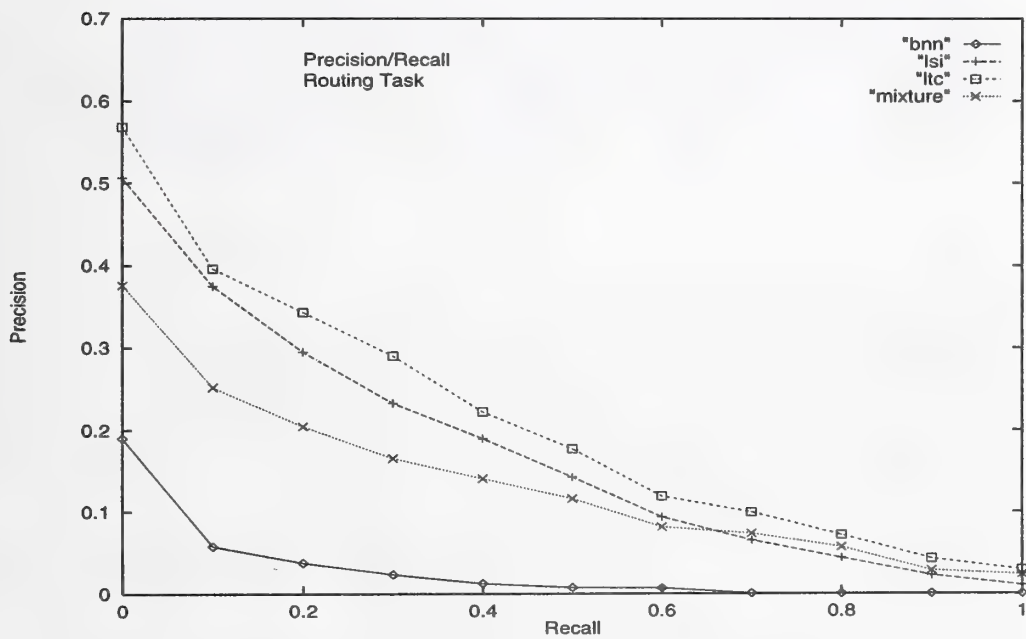


Figure 4: Performance on the Routing Task

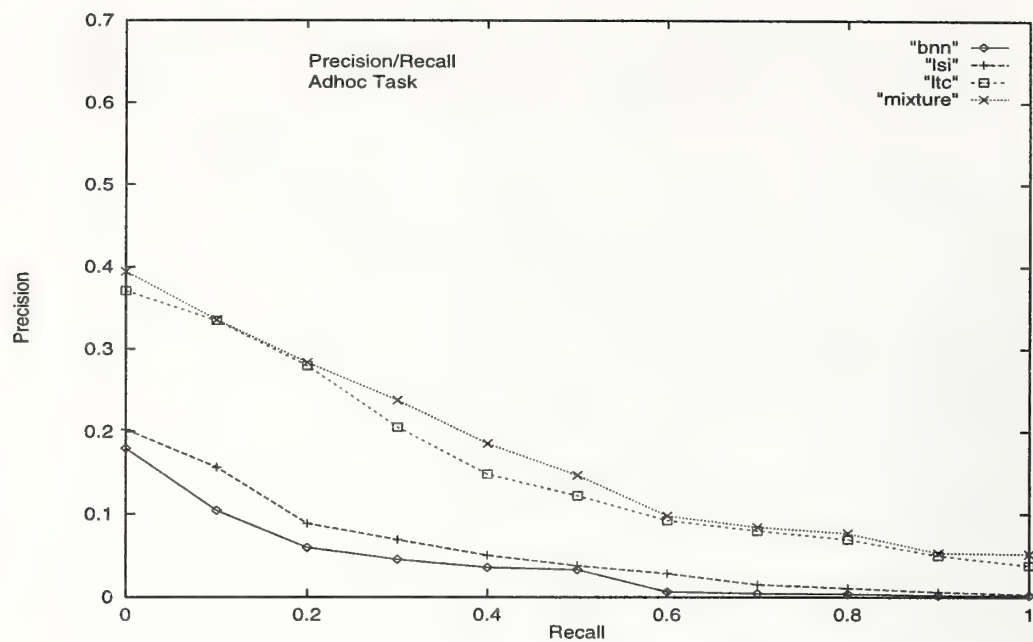


Figure 5: Performance on the Adhoc Task

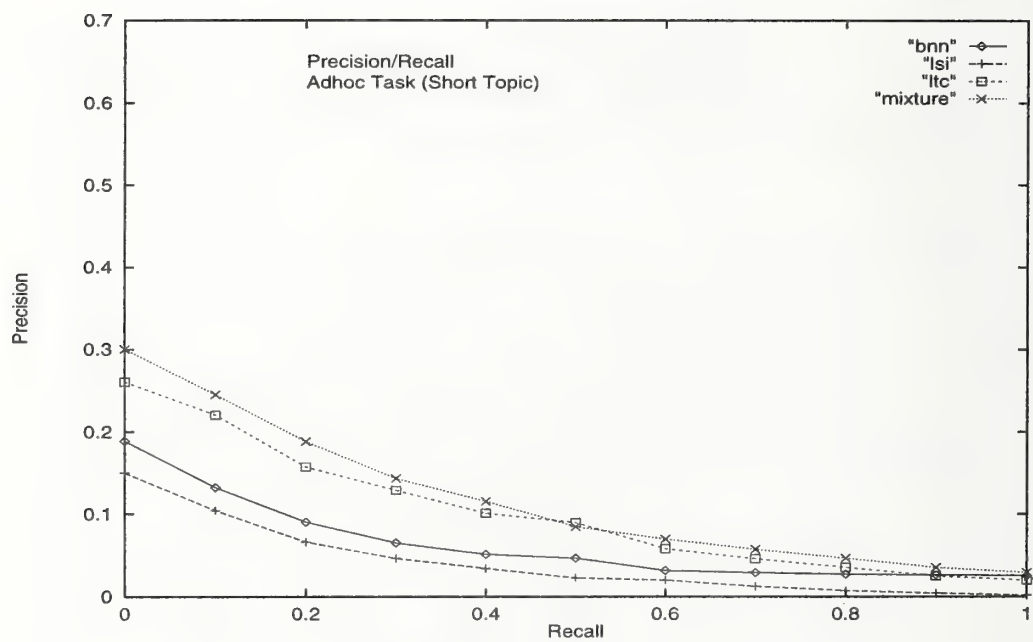


Figure 6: Performance on the Adhoc Task (short-topic)

the mixture outperforms *all* of the individual experts (Figures 5,6). However, for the routing task, the mixture model significantly underperforms two of the three experts (Figure 4).

4 SIMULATIONS

In order to better understand our model and when it can produce improved performance, we performed a series of simulations. First, 1000 pairs of "experts" were generated. An "expert" in this case is simply a list of 1000 documents for which scores have been randomly generated according to a normal distribution. For each pair of experts, we found an approximation to the "best" linear combination of the two by doing a raster scan of both weights in $(0, 1]$ in increments of one tenth, and taking the combination which had the highest average precision when 13 documents were randomly chosen as relevant.

Since our goal was to be able to predict *when* linearly combining the scores of IR systems can result in improvement, we collected some metrics about each pair of experts and used these as the independent variables in a linear regression with "percentage improvement of average precision over the better expert" as the dependent variable. The metrics we chose included: average precision for both experts (p_1, p_2), J for both experts (J_1, J_2), the Guttman's Point Alienation between the two experts (CPA)¹, and the CPA using only the relevant documents (CPA_r). The regression showed that by using only two of the variables we can account for over 80% of the variance in the improvement scores. Those two variables are J_2 (J for the worse of the two experts) and CPA_r (CPA calculated using only relevant documents), with J_2 being weighted slightly more. The importance of both metrics seems intuitively correct: since J_2 is one measure of how well the worse expert performs, it makes sense that an improvement in the better expert would be limited if J_2 was low. Likewise, since CPA_r is a measure of how similar the two experts rank the relevant documents, combining two experts with a very low CPA_r would result in a more random ranking.

5 DISCUSSION

In light of the above simulations, Table 3 shows CPA_r and J_2 for the TREC tasks for which we presented results above. Since the simulations were done on pairs of experts, each possible pairing is shown for each task. Recall that in the simulations, we used CPA_r and J_2 to predict the *best possible* improvement. In the table, we show what the linear regression predicts as the best possible improvement for the actual experts we used for the TREC tasks. Note that for almost every possible pairing in each task, a positive improvement is possible *except* in the routing task.

¹The GPA can be calculated for any two lists of scores x and y as:

$$GPA = \frac{\sum_i \sum_j (x_i - x_j)(y_i - y_j)}{\sum_i \sum_j |x_i - x_j||y_i - y_j|}$$

GPA is a measure of how similar two rankings are to each other. Note that J is simply the GPA between an IR system and a user's relevance judgements, averaged over all queries.

Task	Pair of Experts	CPA_r	J_2	Best Possible Improvement (%)
Training (top 100)	lsi-bnn	0.20	0.35	8.8
Training (top 100)	ltc-bnn	0.11	0.32	7.5
Training (top 100)	ltc-lsi	0.54	0.32	11.1
Training (all)	lsi-bnn	0.22	0.11	3.4
Training (all)	ltc-bnn	0.21	0.11	4.1
Training (all)	ltc-lsi	0.57	0.29	10.6
Adhoc	lsi-bnn	-0.06	0.16	2.7
Adhoc	ltc-bnn	-0.04	0.16	2.8
Adhoc	ltc-lsi	0.50	0.20	8.2
Adhoc (short)	lsi-bnn	-0.03	-0.03	-0.8
Adhoc (short)	ltc-bnn	-0.10	0.18	2.8
Adhoc (short)	ltc-lsi	0.30	-0.03	2.0
Routing	lsi-bnn	-0.04	-0.37	-7.8
Routing	ltc-bnn	0.05	-0.37	-7.0
Routing	ltc-lsi	0.47	0.38	11.7

Table 3: CPA_r , J_2 and Predicted Improvement in Average Precision for Different TREC Tasks

This is precisely the task for which our actual mixture performed the worst. The predicted degradation on the routing task is due entirely to the largely negative value for J_2 , which corresponds to the bnn expert. Thus, because of the poor performance of the bnn expert on the routing task, it seems that improved performance isn't even possible using the simple linear model. It is not clear why the bnn expert has such a low J , but we suspect it is because a large number of terms in the FBIS corpus were not in the training corpus and were ignored. The ltc and lsi experts could make up for the missing terms due to their more sophisticated weighting schemes, but since the bnn expert weights each term equally, each missing term would significantly affect performance.

Table 4 shows the actual improvements we achieved with our mixture, and points to the shortfalls of predicting the best possible performance. We see that whereas the prediction for the Training (all) task is positive, we actually see a significant degradation. Also, the maximum predicted improvement for the Adhoc (short) task is much less than what is actually achieved. Thus, whereas the simulations and the corresponding regression model are useful in explaining the routing task performance, they cannot answer all questions concerning this type of linear mixture model.

The poor performance on the entire available training data (Training (all)) points to the possibility that our technique of only training on the top 100 documents may not scale up from smaller databases to the TREC corpus. This could be due to a number of reasons: the number top-ranked documents that we used may not be representative of the whole training set, or we may have overtrained, or our model just may not be powerful enough to reliably ensure improvement.

Task	Actual Improvement (%)
Training (top 100)	13.1
Training (all)	-31.4
Adhoc	8.4
Adhoc (short)	15.1
Routing	-35.6

Table 4: Actual Improvement in Average Precision for Different TREC Tasks

Task	Expert	Prec	J
Training (top100)	bnn	0.41	0.42
Training (top100)	lsi	0.39	0.35
Training (top100)	ltc	0.38	0.32
Training (all)	bnn	0.07	0.11
Training (all)	lsi	0.15	0.29
Training (all)	ltc	0.23	0.45
Adhoc	bnn	0.04	0.16
Adhoc	lsi	0.06	0.20
Adhoc	ltc	0.16	0.53
Adhoc (short)	bnn	0.06	0.30
Adhoc (short)	lsi	0.04	-0.03
Adhoc (short)	ltc	0.10	0.18
Routing	bnn	0.03	-0.37
Routing	lsi	0.18	0.38
Routing	ltc	0.21	0.45

Table 5: Average Precision and J for Different Experts and TREC Tasks

Our previous work has shown that J and average precision are highly correlated, a finding which justifies optimizing J . Table 5 shows these two measures for the different experts on different tasks. A linear regression of the two measures shows an $r^2 = 0.35$. However, because the Training (top 100) precision was measured using a set of documents with a higher percentage relevant documents, it is artificially inflated and should not be included in the regression. When it is left out, $r^2 = 0.62$, a more reasonable amount of correlation.

6 CONCLUSIONS and FUTURE WORK

We have shown that a simple linear combination of scores from different IR experts can possibly improve the performance of those systems on novel documents. However, such improvement is not guaranteed, and depends on the performance of the individual experts as well as how similarly each is to the other. The model we present is a simple one, yet generally applicable to any collection of IR systems since it does not require knowing about how the systems work, it just needs their scores.

We also present a way of choosing parameters for any IR system model which is independent of the details of the model itself: optimization of a criterion (J) which is correlated with average precision. Despite our mixed results, we still believe that optimizing J is a good way to adjust IR system parameters. We believe our mixed results are artifacts of the way we trained the system and strength of our model.

We intend to further analyze our results from this TREC while preparing for the next one. We will try using a better training regime, one which uses a better optimization technique along with cross-validation to avoid overtraining and to choose the right number of top-ranked documents to train on. We are also in the process of extending our simulation experiments, using more realistic "experts" with reasonable levels of precision and also allowing negative combination weights. For next year's conference, we hope to use a more sophisticated model – one which varies the weights on each expert according to the current document or query. We also hope to be able to use other participants' entries in order to show that any collection of experts can be successfully used.

References

- [Bartell, 1994] Bartell, B. T. (1994). *Optimizing Ranking Functions: A Connectionist Approach to Adaptive Information Retrieval*. PhD thesis, Department of Computer Science & Engineering, The University of California, San Diego, CSE 0114.
- [Bartell et al., 1994a] Bartell, B. T., Cottrell, G. W., and Belew, R. K. (1994a). Automatic combination of multiple ranked retrieval systems. In *Proceedings of the ACM SIGIR*, Dublin.
- [Bartell et al., 1994b] Bartell, B. T., Cottrell, G. W., and Belew, R. K. (1994b). Optimizing parameters in a ranked retrieval system using multi-query relevance feedback. In *Proceedings of the Symposium on Document Analysis and Information Retrieval*, Las Vegas.
- [Berry, 1992] Berry, M. W. (1992). Large scale singular value computations. *International Journal of Supercomputer Applications*, 6(1):13–49.
- [Deerwester et al., 1990] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.
- [Dumais, 1995] Dumais, S. T. (1995). Latent semantic indexing (LSI): TREC-3 report. In [Harman, 1995a].
- [Harman, 1995a] Harman, D., editor (1995a). *The Third Text REtrieval Conference (TREC-3)*, Gaithersburg, MD. National Institute of Standards and Technology. NIST Special Publication.
- [Harman, 1995b] Harman, D. K. (1995b). Overview of the Third Text REtrieval Conference (TREC-3). In [Harman, 1995a].
- [Jordan and Jacobs, 1994] Jordan, M. I. and Jacobs, R. A. (1994). Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6:181–214.
- [Kantor, 1995] Kantor, P. B. (1995). Decision level data fusion for routing of documents in the TREC3 context: A base case analysis of worst case results. In [Harman, 1995a].

- [Knaus et al., 1995] Knaus, D., Mittendorf, E., and Schäuble, P. (1995). Improving a basic retrieval method by links and passage level evidence. In [Harman, 1995a].
- [Salton, 1971] Salton, G., editor (1971). *The SMART Retrieval System - Experiments in Automatic Document Retrieval*. Prentice-Hall Inc., Englewood Cliffs, N.J.
- [Salton and Buckley, 1988] Salton, G. and Buckley, C. (1988). Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 24:513-23.
- [Shaw and Fox, 1995] Shaw, J. and Fox, E. (1995). Combination of multiple searches. In [Harman, 1995a].



Report on the Glasgow IR group (glair4) submission

Mark Sanderson & Ian Ruthven

Department of Computing Science
University of Glasgow
Glasgow G12 8QQ, UK

1 Introduction

This year's submission from the Glasgow IR group (glair4) is to the category B automatic ad hoc section. Due to pressures of time and unexpected complications, our intended application of a technique known as generalised imaging [Crestani 95] was not completed in time for the TREC deadline. Therefore, the submission is the output of an IR system running a simplistic retrieval strategy, similar to last year's submission though with some modifications. It would appear from comparison with other category B submissions that this strategy is relatively successful.

The following sections of this report contain a description of the retrieval strategy used, an analysis of the results, and finally, a discussion of our intentions for TREC 6.

2 Methodology

The retrieval strategy used was a 'text book' approach. The words of the collection and query documents had their case normalised. Any words appearing in a stop list (the creation of which is described below) were removed. The remaining terms were applied to Porter's stemming algorithm [Porter 80]. Document terms were weighted using a *tfidf* scheme as shown in Equation 1, taken from [Crestani 95]. A document was scored in relation to a query by summing the weights of those query terms found within it.

$$w_{ij} = \frac{\log(freq_{ij} + 1)}{\log(length_j)} \cdot \log\left(\frac{N}{n_i}\right) \quad (1)$$

w_{ij} = *tfidf* weight of term *i* in document *j*

$freq_{ij}$ = frequency of term *i* in document *j*

$length_j$ = number of unique terms in document *j*

N = number of documents in collection

n_i = number of documents term *i* occurs in

The stop word list was chosen after a short study of retrieval effectiveness when different lists were used. Those tried were no stop list, the stop list found in Van Rijsbergen [Van Rijsbergen 79], and a list composed of words whose frequency of occurrence, within the document collection being retrieved from, is greater than some level. A number of frequency of occurrence levels were examined, that which was found to produce the highest effectiveness was composed of words that occurred in more than 7.5% of the document collection. As can be seen in

Figure 1, in comparison with the other stop lists, this type of list produced the best effectiveness, therefore it was used in this year's TREC submission.

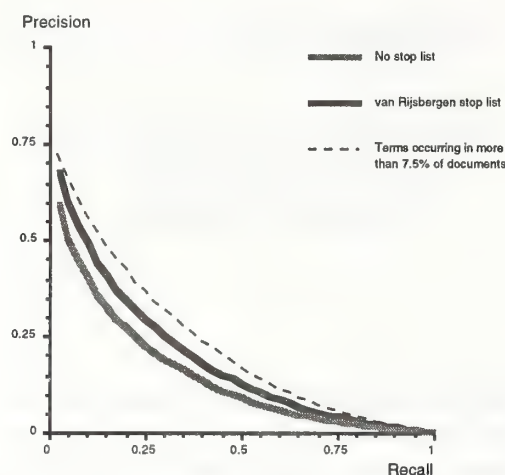


Figure 1. Comparison of retrieval effectiveness using different stop lists.

3 Results

Perhaps surprisingly for such a simplistic retrieval strategy, the glair4 submission appears to have been above average, when compared to the other thirteen category B submissions. Comparing average precision, glair4 was higher than the median average precision in 28 of the 45 topics, equal to the median in 12 topics, and worse in 5.

4 Conclusions and future work

One could say that the relatively good performance of glair4 is a little disheartening. In using a simple retrieval strategy, one might have expected it to perform worse than it did. Nevertheless, as good as this performance appears to be, a comparison is not being made with the best performing systems in the category A ad hoc section. Therefore, next year it is intended that the simplistic strategy described here will be applied to this larger task.

The work to successfully implement the theoretical approach of generalised imaging on an IR system will continue. Last year, limitations of computational resources was a problem; this has been solved by the purchase of new equipment. It has been found, however, that the initial implementation of this approach requires attention and in the next year an altered implementation will be pursued.

5 References

Crestani 95

- F. Crestani, M. Sanderson, I. Ruthven & C.J. van Rijsbergen (1995). The troubles with using a logical model of IR on a large collection of documents, proceedings of the TREC-4 conference.

Frakes 92

W.B. Frakes & R. Baeza-Yeates (1992). Information Retrieval: Data structures & algorithms, in Prentice Hall

Porter 80

M.F. Porter (1980). An algorithm for suffix stripping, in Program - automated library and information systems, 14(3): 130-137.

Van Rijsbergen 79

C.J. van Rijsbergen (1979). Information retrieval (second edition), in London: Butterworths.



Metric Multidimensional Information Space

Gregory B. Newby¹

Graduate School of Library and Information Science
University of Illinois at Urbana-Champaign

The rationale and methodology for retrieval based on the relative locations of documents within a geometric information space are introduced. Results from category A routing and filtering experiments in TREC-5 are discussed. The techniques used are related to the vector space model, latent semantic indexing, and other methods that rely on statistical qualities of texts to assess document relatedness. Results show some promise, but additional research is needed to determine the extent to which retrieval may be improved over existing approaches.

INTRODUCTION

Spatial models for IR are well-known, with almost 30 years of refinement and variation. The primary points of departure of this work from many of the efforts utilizing the vector space model (VSM) or derivatives are:

1. Relations among terms are measured. This is contrary to the basic VSM, in which term vectors are mutually unrelated (orthogonal); and
2. Only a relatively small subset of available terms are utilized.

The first point is the most important; the second is more a matter of computational speed than of desire. The genesis of the specific methodology is based on a technique for the measurement of psychological phenomena through surveys called multidimensional scaling or MDS (KIM78, WF80). This technique results in a third departure from VSM:

3. The agreement between the information space and human cognitive space is subject to empirical verification.

This third quality will not be pursued in this paper, apart than to present the definitions of cognitive space and information space.

Information space is a set of concepts and relations among them held by an information system. Information space is produced by a set of known procedures, and is changed through intentional manipulation of its content.

¹GSLIS, UIUC; 501 East Daniel Street; Champaign, IL; 61820; USA. Tel: 217-244-7365; Email gbnewby@uiuc.edu.

Cognitive space is the set of concepts and relations among them held by people or groups. Cognitive space is in constant change as a result of internal processes and interaction with the environment.

The nature of the cognitive approach to information systems is discussed extensively in INGW96. For this paper, it is sufficient to comment that the desire to develop methods for information space which approximate, in some ways, cognitive space is accepted as worthwhile. An example of an information system which would presumably benefit from such methods would be a customized information agent. For example, a computer program that selects items for presentation to a user from a constant stream of new documents based on *a priori* data on a particular user or group (such a program is indicated for the TREC-5 filtering task).

GENERAL METHOD

Retrieval from the information space as created here is based on the monotonic distance of documents from a query. A multidimensional information space is created based on statistical relations among terms; documents are then located at the centroid of their terms. Queries are also located at the centroid of their terms. Details on the steps involved follow.

Step 1: Select terms for the information space. The space will be built from a statistical analysis of term relations. Rather than assess relations among all unique terms in a set of data, it is desirable to identify a smaller number of terms to work with. For TREC-5, up to 2200 terms were used. Later improvements in programs allowed for up to 4000 terms. Further improvements are anticipated to yield capability for at least 10000 terms with current technologies. Throughout, all terms consist of only one word. It is anticipated that the use of multiple-word terms would yield interesting and useful results, but such an investigation has not yet been made. In addition, the statistical relations employed here could be applied to more complex "terms," such as the conceptual hierarchies described in SC95.

Selecting terms may be accomplished automatically or by consideration of some criteria. An automatic method might be to identify terms which are "frequent, but not too frequent." For example, between the 1st and 3rd quartile in the frequency distribution. (This presumes that a frequency distribution for the occurrence of terms has been calculated for a document collection or a representative subset.)

A non-automatic method for term identification might be to use terms from a collection that have been previously identified as being of interest. For example, a personal filtering system to seek network news (Usenet) articles of interest might use terms from documents that were judged to be of interest in the past. At this phase, a stoplist may be employed to de-select terms that are deemed of little value in discerning differences among documents. Stemming or truncation may also be applied.

For the TREC-5 work described here, a 500 word stoplist from SMART was used throughout. The only stemming used was to remove the letter 's' in the last word position. All words were truncated

at 8 characters. It is anticipated that more intelligent stemming techniques could yield somewhat improved results (e.g., POR80).

Step 2: Generate a term by term co-occurrence matrix. The fundamental measure of “relatedness” applied in this method is the tendency of terms to co-occur in documents. Co-occurrence may be operationalized in a variety of ways for natural language documents. The outcome, regardless of the operational method, is an N by N matrix, where N is the number of terms selected in the previous step. The matrix is square and symmetric. Depending on the number of terms and the number of documents used to create the matrix, it may also be sparse (for TREC-5, no matrices were sparse).

The assumption underlying this method is simply that terms that tend to occur together are more closely related than terms that do not tend to co-occur. This is a much less sophisticated way of assessing term relations than is found in other settings. In expert systems, for example, measures of term relatedness include hierarchical relations, syntactic relations, temporal relations, and others (see, for example, PC88). Pathfinder networks use a more complicated measure of association than co-occurrence, yet without assigning semantic meanings to associations (MS99). Lelu (LELU91) applies a neural network.

Law & Whittaker (LW92) use the term ‘co-word’ method to refer to co-occurrence measures, as do PR93 and LEE93. The benefits of measuring simple co-occurrence is that it may be accomplished automatically for large document collections and that it is computationally simple and conceptually uncomplicated.

One way of operationalizing co-occurrence is to count the number of times each pair of terms in the matrix co-occur within a given (natural language) document. Thus, if **TERM(i)** and **TERM(j)** both occur in a document, the co-occurrence score for row **i** column **j** would be incremented, as would its symmetric pair, row **j** column **i**. This method essentially ignores the tendency for some terms to occur more than once in any given document, and also does not take into account the relative proximity of terms within documents.

A more detailed method would be to only increment a co-occurrence score for term pairs that occur within the same sentence, paragraph, document section, etc. Danowski (DAN88) has found that a sliding “window” of +/- 7 terms is effective in measuring term relatedness. Utilizing this method, terms that occur together in the same document would only contribute to the scores in a co-occurrence matrix when they occur closely together in that document. Terms which tend to occur more frequently in documents will have greater raw co-occurrence scores using this type of method.

A further operationalization of the creation of the co-occurrence matrix might be to post-process co-occurrence scores based on the relative frequencies of terms within documents and within the document collection. **tf * idf** is one common means of balancing the contributions of terms by examining their tendency to occur frequently within individual documents versus within a collection.

For TREC-5, the simplest approach was used: the co-occurrence matrix was built based on terms which occur in full-text documents, regardless of term or document frequency. Thus, a value of, say, 200 in the co-occurrence matrix for a given term pair would indicate that there are 200 document

from the TREC-5 collection in which both terms occurred.

Step 3: Extract a term space from the co-occurrence matrix. Principal components analysis (PCA) is applied to reduce the N by N matrix to a multidimensional term space. Principal components analysis is a multivariate statistical method similar to factor analysis (see KIM78 for a discussion). It is intended to identify trends in sets of data with large numbers of variables where none are pre-identified as dependent or independent. Eigenvalues and eigenvectors (which are used here as coordinates within a multidimensional space) are the primary outcomes of PCA.

Procedurally, PCA consists of iteratively identifying eigenvalues in the co-occurrence matrix. (More accurately, PCA first creates a correlation or covariance matrix from the co-occurrence matrix, then performs the eigenvalue extraction.) Pre-written computer functions and procedures for PCA are available for a variety of settings. The programs utilized in TREC-5 for PCA were written in FORTRAN and utilized the IMSL package. Since that time, the programs have been ported to C, but still utilize the IMSL routines **CORR** and **PRINC**.

The first eigenvalue and associated eigenvectors account for the most variance in the overall matrix of any possible linear combination. The second eigenvalue accounts for the most variance left over after the first, and so forth. By definition, a N by N matrix can be represented completely accurately by at most $N - 1$ eigenvalues (for a real symmetric matrix). In most cases, however, there is enough co-variance in the matrix that fewer than $N - 1$ eigenvalues are needed. As the extraction proceeds, the relative variance accounted for by each successive iteration decreases. It is typical, therefore, to only extract a limited number of eigenvalues by stopping with the cumulative variance accounted for reaches a desired threshold, such as 90 or 99%. PCA performed for TREC-5 typically accounted for 99% of the variance with 250-450 eigenvectors for a 2000-term matrix.

Thus, a N by N matrix may result in a D -dimensional space, where D is far less than N . It is equally accurate to speak of the characteristics of the space in terms of eigenvalues and eigenvectors, matrices, or Euclidean geometry. The term space may be useful before documents are incorporated in the next step. For example, automatic query expansion or thesaurus lookups might be based on terms that are closest in the space to a target term.

Step 4: Locate documents in the space. Document locations in the space are calculated by placing them at the centroid of their associated terms. That is, a document is located at the geometric center of all of the N terms in the space that also occur in the document. For example, a document with 1000 unique terms might have 200 that are among the N terms in the space. The document would then be located at the center of the locations of those 200 terms.

This is the same approach used in the VSM. The difference is that the terms are not unit orthogonal vectors. A vector space with N terms would consist of N dimensions or vectors of equal size (typically, a size of one unit), each of which is mutually independent of the others. The difference in the current approach is that each term has a known quantified relation to every other term in the space. From a conceptual standpoint, this makes little sense because it is unreasonable to assert that terms are unrelated, but it eases computational approaches to assume this is the case.

Choices may be made about the specific method for locating the documents. For example, a document might be proportionally closer to terms that occur more frequently in that document by weighting that term more highly.

At the conclusion of this stage, the information space contains terms and documents. It is now ready for retrieval using various methods. It should be pointed out that empirical evidence through user studies (NEW93) has confirmed that one property of the information space is not what users would expect: although distance scores between terms and documents are known in the space, it is not the case that documents are located closely to their associated terms. This makes sense based on the steps taken above: by placing documents at the center of their associated terms, one would predict that a particular document would not be especially close to any of its terms. Users of a visual interface to such an information space, however, made the (reasonable) assumption that once a useful term in the space had been identified, all they needed to do was retrieve documents close to that term.

METHOD: TREC-5 DOCUMENTS

The information space created by the steps above is dependent on the terms selected for inclusion in the space, and by the documents used to create the co-occurrence matrix of those terms. The overall plan was to make a "context" information space for separate document collections. For example, a space could be built from the ZIFF collection (about 57K separate documents), and a separate space built from the Congressional Record collection (about 28K documents). Different sets of "interesting" terms could be derived from examining the frequency distribution of terms within each collection.

Each query (queries 251-300) could then be located in the separate information spaces and used to retrieve the closest documents to the queries in that space. In fact, this approach was used. One problem with the approach is that there is no good cross-space metric to decide how to mix documents from different collections. For example, in the ZIFF space the closest 100 documents might all be closer than .10 units, while in the Congressional Record space the closest 100 documents are all within .06 units. There's no sound basis, from the methods used here, to decide whether a document at, say, .05 units distance from the query in one space is equally "related" to a document at the same distance in another space. (Note that all the statistical method measures is "relatedness" based on the term co-occurrence matrices — there can be no claim that the distance measures "relevance" or anything else.)

Results of this method were not submitted, however. The author managed to confuse the requirements for the routing and ad hoc tasks, and ended up creating information spaces suitable for ad hoc queries but without realizing that the actual collection to be used was not derived from the "context" spaces. The result was that there was not enough time to complete the ad hoc tasks, nor to generate many separate information spaces. Instead, the process was as follows for both routing and filter tasks, which utilized only one information space:

Select first group terms for analysis. Unique terms from the routing and filtering queries² were determined (minus stoplist terms). 915 unique terms resulted.³ All tagged fields from queries were dropped for all processing discussed here, and the entire query (*sans* tags) was used as-is with no further processing.

Select second group of terms for analysis. Results of the routing and filtering queries from prior TRECs were examined. 5727 of the documents that had been identified as relevant from prior TRECs were included in the collections on CD2 and 4 utilized for TREC-5. A frequency distribution of unique terms from these 5727 documents was created, and an additional 985 terms were selected from approximately the center of the distribution. These terms occurred in between 250 and 4000 of the 5727 documents. These 985 were added to the 915 previously identified, for a total of 1900 terms for further analysis. It is important to stress the non-scientific basis of this selection, as only term frequency was taken into account in order to select a manageable number of terms.

Build the co-occurrence matrix. Only the 5727 documents previously identified as relevant from prior TRECs were used to build the co-occurrence matrix. The idea was to create an information space based strongly on relevance. Again, this is an arbitrary decision: could it not be that the non-relevant documents would have been equally useful? Or, simply using the documents from other collections?

Perform Principal Components Analysis. PCA extracted 99% of the variance from the co-occurrence matrix, resulting in 454 eigenvalues (alternatively, a 454-dimension information space).

Locate documents. Each of the 130476 documents from the FB collection (the target for both the routing and filtering tasks) was located in the 454D space. All terms in each document that were also among the 1900 terms for analysis were identified. Documents were assigned coordinates at the (non-weighted) centroid of the terms.

Locate queries. The routing/filtering queries were located using the exact same technique as for documents, with the same pre-processing. Note that no query expansion techniques were utilized.

Select documents for "retrieval." Results for each query were generated. The geometric distance from each document to each query was measured. A ranked list of all 130476 documents for every query resulted.

Prepare result sets. The queries and collections were the same for the routing and filtering tasks.

²That is, queries with these numbers: 1, 3, 4, 5, 6, 11, 12, 23, 24, 44, 53, 54, 58, 68, 77, 78, 82, 94, 95, 100, 108, 111, 114, 118, 119, 121, 123, 125, 126, 142, 154, 161, 173, 185, 187, 189, 192, 193, 194, 195, 202, 207, 211, 221, 222, 224, 228, 237, 240 and 243.

³Throughout, documents and queries were pre-processed to remove all punctuation, change to lower-case only, remove trailing 's,' and keep only alphabetical characters (remove symbols and numbers). The impact of these hatchet techniques is difficult to assess without further experiment.

For routing, the closest 1000 documents for each query were submitted as results. For filtering, there were three sets. In the task guidelines, these were presented as: "Run 1 should be a high precision run, Run 3 a high recall run, and Run 2 somewhere in between."

Since, as mentioned earlier, the information space only measures "relatedness," a further arbitrary decision was made that Run 1 should be smaller, Run 2 larger, and Run 3 largest. Thus, the closest 25 documents were presented for Run 1, the closest 250 for Run 2, and the closest 500 for Run 3.

RESULTS

Performance of the information space techniques as presented here were fair, but far from outstanding. TREC-5 results are labeled "ISpace" (for lack of a better acronym).

Table 1: ISpace routing results compared to median relevant retrieved @ 1000

Below median	34
On median	6
Above median	5

ISpace seemed to do well at including relevant documents at the median or above when the total judged relevant was small (i.e., < 10). Overall, at least ½ of the total relevant documents were retrieved as part of the response set. Otherwise, there is no obvious pattern to the results.

Table 2: Summary of routing results across queries

Query number	Relevant	Relevant retrieved	Precision at 10 docs	Precision at 100 docs	Precision at 1000 docs	Exact Precision
1	30	15	.10	.010	.015	.033
3	101	26	.00	.01	.026	.0099
4	178	105	.10	.03	.105	.0281
5	19	13	.00	.00	.01	.00
6	158	49	.00	.02	.049	.0190
11	92	78	.00	.00	.078	.00
12	228	171	.00	.06	.17	.0658
23	7	4	.00	.00	.004	.00

24	38	16	.00	.00	.016	.00
44	10	6	.00	.00	.006	.00
53	1	1	.00	.00	.001	.00
54	48	38	.00	.01	.038	.028
58	45	12	.00	.00	.012	.00
77	14	7	.00	.01	.007	.00
78	37	21	.00	.01	.0210	.00
82	55	49	.00	.00	.049	.00
94	56	17	.00	.00	.017	.00
95	93	23	.00	.00	.023	.00
100	157	79	.00	.02	.079	.0127
108	174	101	.00	.12	.101	.0977
111	887	458	.60	.62	.458	.4307
114	42	14	.00	.01	.014	.0238
118	324	23	.0255	.00	.00	.0015
119	185	13	.00	.03	.013	.0216
123	57	22	.00	.02	.022	.0175
125	6	3	.00	.00	.003	.00
126	18	11	.00	.01	.011	.00
142	808	450	.50	.65	.45	.3837
154	22	7	.00	.00	.007	.00
161	153	26	.00	.03	.026	.0261
173	15	5	.00	.00	.005	.00
175	14	10	.00	.00	.01	.00
187	194	82	.00	.03	.08	.0206
189	584	264	.30	.37	.264	.1884
192	10	6	.00	.00	.006	.00
194	8	3	.00	.01	.003	.00

202	583	250	.50	.50	.25	.247
207	1	1	.00	.00	.001	.00
211	2	0	.00	.00	.00	.00
221	193	32	.00	.02	.032	.0311
222	2	2	.00	.01	.002	.00
224	1	0	.00	.00	.00	.00
228	68	43	.00	.02	.043	.0294
240	88	11	.00	.00	.011	.00
243	2	2	.00	.01	.002	.00

Precision ranged from extremely poor (exact values of 0 for 25 queries) to nearly acceptable (over .20 for 5 queries). Queries for which performance was acceptable include 111, 142, 189 and 202.

Table 3: Results over all 45 topics:

Query	Relevant	Relevant Retrieved	Precision at 10	Precision at 100	Precision at 1000	Exact Precision
all 45	1403	542	.05	.05	.03	.0299

Filtering

Filtering task results, at first glance, were not much different than the results for the routing task. Of the 7 groups that tackled this task, though, Ispace would seem to have performed the most poorly across the board. Of the 45 queries, Ispace achieved the lowest precision score on all three sets (precision oriented; balanced; recall oriented) for 20. That is, none of the other groups received a lower score for any of the those 20 queries.

For the remaining 25 queries, Ispace achieved the lowest score in at least one of the three sets for an additional 18 queries. Ispace had the highest score for a set for only 2 queries, and achieved a score close to the median on at least one set for an additional 4 queries.

An easy explanation for the poor performance of Ispace on many queries is forthcoming. In queries that resulted in very few relevant documents overall, the fixed number of documents retrieved resulted in very low scores for the three sets. 13 queries resulted in 10 or fewer documents judged as relevant. Each of these queries were among the 20 on which Ispace had the lowest achieved score.

Table 4: Summary filtering task results

	Precision oriented	Balanced	Recall-oriented
Mean precision	.2347	.1065	.0788
Mean recall	.0958	.3008	.3716

Analysis of the global statistics for all routing and filtering tasks indicates that other groups may have also had troubles due to the relatively small number of relevant documents found for the queries. Note that mean number of relevant documents per query is 129.1, but the standard deviation is 204.1. 15 queries had 20 or fewer relevant documents. This low number of relevant documents would result in low precision scores for systems which produced a ranked list of 1000 documents per query for the routing results, or who chose (as was the case for Ispace) to produce fixed-sized sets for the filtering tasks.

Table 5: Summary of routing and filtering task results for all of TREC-5 compared to Ispace

	Mean	Standard Deviation	Sum
Relevant	129.1	204.1	5808
Rel_Ret by Ispace	57.08	104.6	2569

Correlation Analysis

Analysis of Pearson product moment correlation coefficients across the set of scores produced for Ispace's routing results is slightly informative. Coefficients tend to be over .90 for almost all score pairs when all scores are compared (scores for relevant, rel_ret, recall at .00 - 1.0, precision at 5 - 1000 documents, average precision, and exact precision). This would be expected, since most scores rely on rel_ret. The only interesting feature of the correlation table (not presented here) is that correlations tend to drop off for recall after the .50 level, with many non-significant or small values. One interpretation of this is that the cluster of documents that Ispace places near the query is effective, but only to a point. Later-retrieved documents (documents further from the query location) are less likely to be relevant, yet other relevant documents *do* exist in the information space.

It may be inferred from this tendency that the measure of "relatedness" that Ispace is sensitive to is not entirely consistent with the measure of "relevance" as produced by the TREC-5 evaluators. Of course, that can be said of any system that does not achieve perfect results! For Ispace, an interesting question is whether the relevant documents not found close to the query location tend to cluster together in other locations in the information space.

Queries

An analysis of the queries themselves may be illuminating. Factors which might contribute to the effectiveness of Ispace include the length of queries, the presence of query terms in the documents of the information space, and the number of relevant documents from prior TRECs that were used to build the information space.

From the method as described herein, only one global information space was built for the routing and filtering tasks. The input to the space was the collection of documents that had been identified previously as relevant and were part of the CD2 or CD4 collections.

Table 6: Summary statistics for numbers of query terms and relevant documents used to build the information space

Figures for 45 TREC-5 routing/filter task queries	Number of terms in each routing query	Number of relevant document per query used for training from prior TRECs	Total relevant docs found for all TREC-5 routing/filter participants	Ispace total relevant docs for routing tasks
Mean	120.7	442.8	129.1	57
Median	116	372	48	16
Minimum	8	66	2	0
Maximum	522	1371	887	458
Standard Deviation	89	276.9	204.1	104.6

The figures in table 6 demonstrate variety in query sizes, and also that overall, Ispace seems to be capable of retrieving about ½ of the relevant documents. One item which is strongly supported is that the number of query terms is not related to the number of documents identified by TREC-5 evaluators as relevant ($p > .10$), nor is the number of query terms related to the number of relevant documents retrieved by Ispace ($p > .10$).

Information Space Input Documents

The interesting, but inconclusive, component of the correlation table of query length and number of documents retrieved and ranked as relevant is that there is a strong relationship between the number of documents that went into building the information space from each query and the number of relevant documents that Ispace eventually retrieved ($r=.57$, $p < .001$). Taken alone, one would infer that the context of the information space is extremely important — that including the relevant documents from past experience is very useful in improving results.

But the high correlation between the number of documents that went into building the information space and the number of relevant documents retrieved overall ($r=.61$, $p < .001$) indicates a different interpretation. Essentially, this correlation indicates that queries with larger numbers of relevant documents in past TRECs tended to have larger numbers of relevant documents in TREC-5.

Table 7: Correlation among values for TREC-5 and Ispace

<i>Pearson's r</i> , probability	Relevant docs prior TRECs	Relevant docs TREC-5	Relevant docs by Ispace	# post-processed query terms
Rel docs prior	1.0 0.000			
Rel docs TREC-5	0.614 0.001	1.0 0.000		
Rel docs Ispace	0.568 0.001	0.954 0.001	1.0 0.000	
# query terms	0.172 0.257	0.136 0.373	0.123 0.417	1.0 0.000

Examination of the queries that Ispace was able to perform best at helps to give sustenance to the first interpretation, that the context used to build Ispace's information space improves retrieval for queries more strongly related to or derived from that context. The queries on which Ispace did best for both filtering and routing tasks are 111, 142, 189, and 202.

Examination of these queries shows they are unremarkable in terms of the number of query terms (ranging from 8 to 189). Yet two of the queries (142 and 189) had the largest number of relevant documents from prior TRECs that went into the context for Ispace's information space. The other two queries (111 and 202) were also above the 3rd quartile.

The greatest sustenance is given to the second interpretation, however — that Ispace tends to retrieve larger numbers of relevant documents when all of TREC-5 identified larger numbers. This is the correlation of .95 ($p < .001$) between the number of relevant documents identified by TREC-5 evaluators and the number of relevant documents that Ispace retrieved.

Examining the number of relevant documents from prior TRECs that were actually on CD2 and CD4 (as opposed to those that were judged as relevant in prior TRECs but not part of the collection used for TREC-5) is not helpful, as the proportion of documents judged relevant to those on CD2 and CD4 is relatively constant. However, a somewhat weaker correlation is found between the number of relevant documents Ispace found and the number of documents actually used to build the information space ($r=.47$, $p < .001$). The weaker correlation (.47 versus .56 for all relevant documents from prior TRECs) is just as likely to be a byproduct of the larger standard deviation for the second score (277, versus 105 for documents actually used) as anything else.

CONCLUSION

Taken together, these results indicate there is some utility in the Ispace approach. However, it is yet to be seen whether the differences in the approach to document processing, document representation, query representation, and the retrieval process from Ispace offers any substantial improvements over existing approaches.

Shortcomings of creating a metric multidimensional information space are primarily practical: the large matrices and moderate computational complexity of the principal components analysis, coupled with the pre-processing of documents and then locating them in the information space, all result in a process that is not amenable to real-time production. By sampling from the documents to create the information space (rather than examining all), and by applying singular-value decomposition or another reduction technique to reduce the size of the matrix, some speed-up in processing would be expected.

Further research should address, minimally, the following areas:

1. What difference in performance would result from adjusting the selection of input documents to generate the information space as follows:

- Choose a small set of known relevant documents only
- Choose a large random set of documents
- Manually pick terms for the space which are suspected to be useful for discriminating among relevant and non-relevant documents

2. What is the effect of the number of terms in the information space (or how they are selected)?

3. Examine how documents identified as "relevant" for TREC-5 tend to cluster in the information space. A random pattern would indicate that there are qualities of relevance that are entirely missed by Ispace, while the presence of well-defined clusters that are not at the query centroid would indicate that Ispace is sensitive, but the query location is not the only worthwhile basis for retrieval.

4. Create information spaces based on actual measurement of cognitive space (i.e., through measurement via MDS surveys), and determine:

- The differences in the information spaces that result for different user groups or situations
- Whether such information spaces could be approximated algorithmically

- Whether better retrieval results may be obtained with such information spaces

Many other areas are of interest, such as the role of multi-word terms, exploration in the effects of different stemming and truncation, or use of knowledge about different parts of documents (e.g., examining whether using only an abstract would help). One area which has not been explored in the current paper is the potential for experimentation with visual interfaces for IR.

The information spaces created here are different from vector spaces in that the term vectors are non-orthogonal. Further, the PCA procedure that creates the spaces extracts eigenvalues such that the largest are chosen first. Thus, it is typical for the first 3 eigenvalues to account for 15 or 20% of the variance in the information space. These first 3 dimensions, then (or, another set of early dimensions) are suitable for visualization and navigation with a 3D interface.

The author has created such interfaces using flat-screen and 3D technologies, and has found them to be usable for actual retrieval (NEW92). Such an interface is not well-suited to most of the TREC-5 tasks, but could be employed for the interactive task. Such efforts remain for a future time.

REFERENCES

- [DAN88] Danowski, James A. 1988. "Organizational infographics and automated auditing: Using computers to unobtrusively gather as well as analyze communication." in Goldhaber, G.M. & Barnett, G.A. (Eds.). Handbook of Organizational Communication. Norwood, NJ: Ablex.
- [ING96] Ingwersen, Peter. 1996. "Cognitive perspectives in information retrieval interaction: Elements of a cognitive IR theory." J. Documentation 52(1): 3-50.
- [KIM78] Kim, Jae-On & Mueller, Charles W. 1978. Factor Analysis. Beverly Hills: Sage.
- [LW92] Law, J. & Whittaker, J. 1992. "Mapping acidification research: A test of the co-word method." Scientometrics 23(3): 417-461.
- [LEE93] Lee, Joon Ho; Kim, Myoung Ho; & Lee, Yoon Joon. 1993. "Information retrieval based on conceptual distance in is-a hierarchies." J. Documentation 49(2): 188-207.
- [LELU91] Lelu, Alain & Claire, Francois. 1992. "Information retrieval based on a neural unsupervised extraction of thematic fuzzy clusters." Neuronimes.
- [MS88] McDonald, James E. & Schvaneveldt, Roger W. 1988. "The application of user knowledge to interface design." in Guindon, Raymonds (Ed.). Cognitive Science and its Applications for Human-Computer Interaction. Hillsdale, NJ: Lawrence Erlbaum.
- [NEW93] Newby, Gregory B. 1992. "An investigation of the role of navigation for informatin retrieval." Proc. Amer. Soc. for Information Science Annual Meeting 20-25. Medford, NJ: Learned

Information.

[PC88] Parsaye, * & Chignell, Mark. 1988. Expert Systems for Experts.

[PR93] Peters, H.P.F. & van Raan, A.F.J. 1993. "Co-word-based science maps of chemical engineering. Part I: Representations by direct multidimensional scaling." Research Policy 22: 23-45.

[POR80] Porter, M.F. 1980. An Algorithm for Suffix Stripping. Program 14(3): 130-137.

[SC95] Strzalkowski, Tomek & Carballo, Jose Perez. 1995. "Natural Language Information Retrieval: TREC-4 Report." In: TREC-4 Proceedings. Gaithersburg, MD: National Institute of Science and Technology.

[WF80] Woelfel, Joseph D. & Fink, Edward L. 1980. The Measurement of Communication Processes: Galileo Theory and Method. New York: Academic Press.

Corpus Analysis for TREC 5 Query Expansion

Susan Gauch and Jianying Wang
{sgauch,jwang}@eecs.ukans.edu
Electrical Engineering and Computer Science
University of Kansas

ABSTRACT

Accessing online information remains an inexact science. While valuable information can be found, typically many irrelevant documents are also retrieved and many relevant ones are missed. Terminology mismatches between the user's query and document contents is a main cause of retrieval failures. Expanding a user's query with related words can improve search performance, but the problem of identifying related words remains.

This research uses corpus linguistics techniques to automatically discover word similarities directly from the contents of the untagged TREC database and to incorporate that information in the SMART information retrieval system. The similarities are calculated based on the contexts in which a set of target words appear. Using these similarities, user queries are automatically expanded, resulting in conceptual retrieval rather than requiring exact word matches between queries and documents.

1. INTRODUCTION

Expanding a user's query with related terms can improve search performance. Relevance feedback systems, where related terms come from the contents of user-identified relevant documents, have been shown to be quite effective (Harman 1992). Our earlier work showed that an expert system which automatically reformulated Boolean queries by including terms from an online thesaurus was able to improve search results (Gauch and Smith 1991; Gauch and Smith 1993) without requiring relevance judgments from the user. Some systems (Anick, Brennan et al. 1990) present related terms to the user and allow them to selectively augment the query. However, the latter two approaches require the presence of an online thesaurus whose words closely match the contents of the database.

Where can such a thesaurus come from? In some cases, it is hand-built (Gauch and Smith 1991), a time-consuming and ad hoc process. In other cases, the thesaurus is an online version of a published thesaurus or semantically coded dictionary (Liddy and Myaeng 1993). However, an online published thesaurus or dictionary will have serious coverage gaps if used for technical domains which have their own distinct sublanguages. Because of ambiguity, this type of thesaurus may also be difficult to use with a database of general English documents because they show all possible classifications for a word when only one or a few senses may be actually present in the database.

Our system automatically discovers related words directly from the contents of a textual database and incorporates that information in the modified SMART information retrieval system. We modified and applied one particular technique from the field of corpus linguistics which seemed particularly well-suited for this task. HNC's MatchPlus system (Gallant, Hecht-Nielsen et al. 1993) has a similar approach, however, they use neural networks to identify features which are used to index documents rather than using the words themselves. In contrast, we index documents by their words and identify related words which can be used for query expansion. With our approach, it is possible to provide query expansion on top of pre-indexed collections.

2. SYSTEM ARCHITECTURE

To incorporate the results of the corpus analysis into an existing retrieval engine, SMART was modified to allow it to expand queries based on the similarity matrices, search the database with the expanded queries, and return the top 1000 documents for each query.

We have modified smart.11.0 at the point when a query is passed to the search engine. At this point, the query words are expanded using different techniques according to the word similarity weights generated from the corpus analysis program. If the databases are treated separately, an algorithm is used to select the appropriate similarity matrix to use to expand each query.

We participated in Adhoc category A, which is evaluated based on the combined collection consisting of the documents on both Tipster Disk 2 and TREC Disk 4. The whole database is 2.1 GB in size and contains 524929 documents from 7 different sources (AP, CR, FR88, FR94, FT, WSJ, and ZIFF). Indexing the database as a whole took approximately 5 hours on a shared Sun SPARC center 2000. The retrieval runs on the combined collection used SMART's "lnc" weights for the documents and the modified "lrc" weights for the queries.

3. CORPUS LINGUISTICS TECHNIQUE

Methods that work with entirely untagged corpora have recently been developed which show great promise (Brill and Marcus 1992; Finch and Chater 1992; Hearst, 1992, Myaeng and Li 1992; Schütze 1992). Using a much more fine-grained approach than traditional automatic thesaurus construction techniques, word-word similarities are automatically calculated based on the premise that words which occur in similar contexts are similar. These techniques are particularly useful for specialized text with specialized vocabularies and word-use, for which there are no adequate online dictionaries. They are also appropriate for general English corpora since a general online dictionary may show many senses for a common word where only one or a few actually are used in a given corpus.

We have modified a corpus linguistics approach (Finch and Chater 1992) that takes into account both the relative positions of the nearby context words as well as the mutual information (Church and Hanks 1990) associated with the occurrence of a particular context word. We have applied this to a 15% sample of the TREC5 database to calculate *a priori* the similarities of a subset of the words in the database, called the target words.

3.1 Similarity Calculation

Similar to (Finch and Chater 1992), the context vector for a word (the *target word*) is a concatenation of the vectors describing the words observed in the preceding two positions and the following two positions (the *context positions*). Each position is represented by a vector corresponding to the occurrence of the 200 highest frequency words in the corpus (the *context words*), giving a 800-dimensional vector describing the context. Initially, the counts from all instances of a word form w_i are summed so that the entry in the corresponding context word position in the vector is the sum of the occurrences of that context word in that position for the corresponding target word form; it is the joint frequency of the context word.

Consider an example in which there are only five context words, {"a", "black", "dog", "the", "very"} and two sentences containing the target word "dog":

- (1) The black dog barked very loudly.
- (2) A brown dog barked very loudly.

Sentence	Context Position	Observed Word	Context Vector
1	-2	"The"	(0, 0, 0, 1, 0) 4th context word
	-1	"black"	(0, 1, 0, 0, 0) 2nd context word
	+1	"barked"	(0, 0, 0, 0, 0) not a context word
	+2	"very"	(0, 0, 0, 0, 1) 5th context word

Table 1. The context vectors for each of the 4 context positions around the occurrence of the target word "dog" in sentence 1.

The context vector for "dog" in sentence 1 is formed by concatenating the context vectors for each of the 4 context positions:

(0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1)

Similarly, the context vector for "dog" in sentence 2 would be:

(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1)

and the combined vector for the word "dog" would be:

(1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2)

Using 150 context words and window size 5, 600-dimensional context vectors are created. Subsequently, 600-dimensional vectors of mutual information values, *MI*, are computed from the frequencies as follows,

$$MI(cw) = \log_2 \left(\frac{Nf_{cw}}{f_c f_w} + 1 \right)$$

This expresses the mutual information value for the context word *c* appearing with the target word *w*. The mutual information is large whenever a context word appears at a much higher frequency, f_{cw} , in the neighborhood of a target word than would be predicted from the overall frequencies in the corpus, f_c and f_w . The formula adds 1 to the frequency ratio, so that a 0 (zero) occurrence corresponds to 0 mutual information. When the mutual information vectors are computed for a number of words, they can be compared to see which words have similar contexts. The comparison we chose is the inner product, or cosine measure, which can vary between -1.0 and +1.0 (Myaeng and Li 1992) .

Finally, to make the identification of the most highly similar terms to a given term more efficient, an auxiliary file is produced *a priori* from the similarity matrix. It stores, for each target word, the words and similarity values for all words with similarity above a given threshold. This is called the similarity lists.

3.2 Preprocessing the Database

For use by the corpus analysis program, the TREC5 databases are sampled randomly and uniformly to get a representative sample (roughly 15%) for each database. Then, the sample database is preprocessed. During preprocessing, the required fields of a document are extracted, tokenized and downcased, and some tokens are treated specially (e.g., U.S. → usa). Then, a sentence tagging algorithm is used to identify sentence breaks. Finally, a word-frequency list for the sample database is generated and sorted by the frequency. Based on this frequency list, target words and context words are automatically selected. Based on earlier experiments, 6000 target words are selected by extending up 500 and down 1500 words from the 4000 word window which captures the maximum number of non-stopped query words. Since only words in the target words will be expanded, any missing important query words are added to the target word file.

The full database goes through a subset of the above steps (downcasing, token replacement) before being indexed by SMART.

4. EXPERIMENTS

4.1 Tuning the calculation on TREC4 category B collection

To tune the corpus analysis program, we ran a series of experiments with TREC4 queries. There are five main parameters which can be adjusted: the sample database size, the list of target words (position and size in the frequency list), the list of context words, the window size (the window around target words that is used to characterize the contexts in which they appear), and the similarity threshold that is used during query expansion. First, we fixed the list of context words (200 highest frequent words in the sample database), the number of target words (4000), and the window size (7). Then, we experimented by shifting the 4000 target words along the frequency list for different sizes of the sample database. We found that the 4000 target words list position that yielded the best 11 point average occurred with a constant ratio of (frequency of first word in target list in sample / (frequency of most frequent word in sample)).

sample size	10%	20%
most frequent word	417969	796667
most frequent target word	120	241
(best threshold, 11-pt avg)	(0.34, 0.1839)	(0.45, 0.1840)
ratio	120/417969	241/796667
	0.0002871	0.0003025
average	(0.0002871 + 0.0003025)/2 = 0.000295	

Table 2. Calculate the starting frequency as percentage of the most frequency (Note: unexpanded queries yield 11-pt avg = 0.1791)

More experiments using the following equation:

$$\text{starting-frequency} = \text{most-frequency} * 0.000295$$

to select 4000 target words were conducted to confirm the above relationship. Table 3 shows the pairs of (similarity threshold, 11-pt avg) for some of the experiments shifting the 4000 target words along the frequency list after using the frequency equation to get the initial siting of the 4000 target words. Note, shifting up N words means shifting the target word window N words more frequent (i.e., selecting N more frequent target words and dropping N less frequent target words). Shifting down N words means selecting less frequent target words. This confirmed that the initial area selected by the ratio performed best. For the following table, 200 context words, a context window of 7 and a 10% sample size were used.

Target-shift	(threshold, 11 pt avg)
1000 up	(0.80, 0.1791)
frequency equation	(0.38, 0.1887)
1000 down	(0.38, 0.1807)
2000 down	(0.27, 0.1818)
3000 down	(0.27, 0.1813)
4000 down	(0.32, 0.1800)

Table 3. Sample size 20% (Note: unexpanded queries yield 11-pt avg = 0.1791)

We evaluated another approach for choosing 4000 target words by selecting the area of the frequency list that contained the maximum number of non-stopped query words. We found that the 4000 target words selected in this manner tended to be shifted up (i.e., more frequent) about 1200 positions relative to the ratio method. Because it was independent of sample size, we decided to use the query words to site the target list, but to shift down 1200 to more closely match the ratio results.

Once the target word list was decided, we did a series of experiments with the list of context words, the window size and the sample database size. From the results in Table 4, we know the parameters with window size 7, context words 200 (the most frequent 200 words) provide better performance. However, the best sample size is not clear, although, in general, the 20% sample size seems to produce the best results.

Sample size 10%			
Context\Window	5	7	9
150	(0.39, 0.1823)	(0.37, 0.1834)	(0.30, 0.1830)
200	(0.39, 0.1825)	(0.34, 0.1831)	(0.27, 0.1815)
250	(0.37, 0.1850)	(0.30, 0.1830)	(0.27, 0.1826)
Sample size 20%			
Context\Window	5	7	9
150	(0.50, 0.1813)	(0.45, 0.1840)	(0.40, 0.1860)
200	(0.45, 0.1834)	(0.38, 0.1887)	(0.34, 0.1873)
250	(0.45, 0.1811)	(0.38, 0.1852)	(0.32, 0.1861)
Sample size 30%			
Context\Window	5	7	9
150	(0.80, 0.1791)	(0.70, 0.1791)	(0.60, 0.1792)
200	(0.70, 0.1791)	(0.60, 0.1796)	(0.40, 0.1804)
250	(0.60, 0.1799)	(0.45, 0.1795)	(0.45, 0.1799)

Table 4. Experimental results for context words, window size and sample size. (Note: unexpanded queries yield 11-pt avg = 0.1791)

4.2 Query Expansion Techniques

If we use all entries in the similarity matrix above a given threshold to expand queries, query words like "identify" and "create" with many strong matches will be greatly expanded while lower frequency query words (usually important query words) don't get expanded at all. Another problem with using the similarity threshold to control the expansion is that minor changes in the similarity threshold can result in large differences in the number of expansion words for certain query words. So, we experimented with an expansion technique based on using a fixed number of expansion words for each query word appearing in the similarity matrix. Table 5 shows the pairs of (threshold, 11-pt Avg) or (max. expanded, 11-pt Avg) for the same similarity matrices with 20% sample size, 200 context words, and 7 window size:

target-words\method	Similarity threshold	max expanded
4000	(0.38, 0.1887)	(1, 0.1802)
6000	(0.80, 0.1791)	(3, 0.1849)
7000	(0.80, 0.1791)	(1, 0.1836)
8000	(0.80, 0.1791)	(3, 0.1834)
9000	(0.80, 0.1791)	(3, 0.1851)

Table 5. Experiments for the size of target words and expanding techniques. (Note: unexpanded queries yield 11-pt avg = 0.1791).

In this experiment, we also evaluated using more target words. In each of the cases, we expand the original 4000 target words by including 1000 more frequent words and then as many less frequent words to make up the total. For example, the list of 7000 target words is extended from the 4000 target words by adding 1000 more frequent and 2000 less frequent words. Although the best overall performance was provided by the threshold expansion technique, it didn't seem robust. For all of the larger target word lists, expansion based on the threshold led to a degradation in the 11-pt average. However, using a fixed number of expansion words seemed much less sensitive. In particular, adding the three most similar words for each query word seemed to consistently improve performance. With this technique, 6000 target words seemed a reasonable compromise between search quality results and time and space complexity.

4.3 Matrix selection for query expansion

Since the whole collection has seven databases (AP, CR, FR88, FR94, FT, WSJ, and ZIFF) for category A, seven similarity matrices are generated, one for each database. We chose to do this rather than using one big similarity matrix for the whole collection because each of the databases has a different domain of interest, resulting in different word usage. Our experiments for TREC4 also confirm that analyzing the databases separately performs better than treating the corpus as one big database. However, it is very important to select the correct matrix for each query. To do this, for each query, we calculate the frequency of each non-stopped query word normalized by the most frequent word in the database over all 7 databases. Then, we sum the normalized frequencies together for all query words. We expand this query using the similarity matrix built from the database which maximizes this sum.

4.4 TREC5 Results

Based on our experiments, we chose a sample size of 15% for TREC5, the corpus is treated as seven separate databases producing seven separate similarity matrices (200 context words, 6000 target words, window size of

7). The queries are expanded using a combination of the threshold and fixed number of words as follows: A maximum of three words are added per query word. However, only words above a certain threshold will be added (0.28 in KUSG2; 0.20 in KUSG3). The following table summarizes our results compared to the entries with short topics in the ad hoc category. Unfortunately, our expanded queries actually performed worse than the unexpanded queries. This was naturally quite disappointing and we have begun trying out different query expansion techniques. One, KU-new, is presented here. It expands only one word for each concept, and only if the similarity value exceeds 0.28. This does, indeed result in improvement, albeit minor.

	No exp.	KUSG2	KUSG3	KU-new
Precision (20 docs)	0.1980	0.1900	0.1860	0.1950
Precision (100 docs)	0.1276	0.1260	0.1246	0.1290
Non-interp. 11 pt avg	0.1100	0.1073	0.1056	0.1127

Table 6. TREC5, category A results.

4.5 Discussion

Our extensive experiments with the TREC4 corpus leads us to believe that our approach can, indeed, lead to improvements in the 11-pt average of greater than 5%. However, we clearly need to come up with a systematic way of setting the parameters for a new database and/or training on a new database in order to achieve performance improvements. This is the focus of our current work.

BIBLIOGRAPHY

- Anick, P.G., Brennan, J.D., Flynn, R.A., Hanssen, D.R., Alvey, B., and Robbins, J.M. (1990). A Direct Manipulation Interface for Boolean Information Retrieval via Natural Language Query. Proc. 13th Ann. International ACM SIGIR Conf. (pp. 135-150). Brussels, Belgium: ACM Press.
- Brill, E., & Marcus, M. (1992). Tagging an Unfamiliar Text with Minimal Human Supervision. In AAAI Fall Symposium Series: Probabilistic Approaches to Natural Language (Working Notes), (pp. 10-16). Cambridge, MA.
- Church, K. W., & Hanks, P. (1990). Word Association Norms, Mutual Information and Lexicography. Computational Linguistics, 16(1), 22-29.
- Finch, S., & Chater, N. (1992). Bootstrapping Syntactic Categories Using Statistical Methods. In W. Daelemans & D. Powers (Ed.), Proc. 1st SHOE Workshop, (pp. 229-235). Tilburg U., The Netherlands.

- Gallant, S.I., Hecht-Nielsen, R., Caid, W., Qing, K., Carleton, J., and Sudbeck, C.D. (1993). HNC's MatchPlus System, 1st Text Retrieval Conf. (TREC-1), (pp. 107-111). NIST #500-207.
- Gauch, S., & Smith, J.B. (1993). An Expert System for Automatic Query Reformulation. J. of the Amer. Society of Inf. Sci., 44 (3), 124-136.
- Gauch, S., & Smith, J.B. (1991). Search Improvement via Automatic Query Reformulation. ACM Trans. on Information Systems, 9 (3), 249-280.
- Harman, D. (1992). Relevance Feedback Revisited. Proc. 15th Ann. International ACM SIGIR Conf., (pp. 1-10). Copenhagen, Denmark: ACM Press.
- Hearst, M.A. (1992). "Automatic Acquisition of Hyponyms from Large Text Corpora," Proc. 14th Intern'l Conf. on Computational Linguistics, Nantes, France, July.
- Liddy, E.D., & Myaeng, S.H. (1993). DR-LINK's Linguistic-Conceptual Approach to Document Detection, 1st Text Retrieval Conf. (TREC-1), (pp. 113-129). NIST #500-207.
- Myaeng, S. H., & Li, M. (1992). Building Term Clusters by Acquiring Lexical Semantics from a Corpus. In Y. Yesha (Ed.), CIKM-92, (pp. 130-137). Baltimore, MD: ISMM.
- Schütze, H. (1992). Context Space. In AAAI Fall Symposium Series: Probabilistic Approaches to Natural Language (Working Notes), (pp. 113-120). Cambridge, MA:
- Chris Buckley (1985). Implementation of the SMART information retrieval system. Technical Report 85-686, Computer Science Department, Cornell University, Ithaca, New York.
- Gauch, S. & Chong, M. (1995) Automatic Word Similarity Detection for TREC4 Query Expansion

Alignment of Spanish and English TREC Topic Descriptions *

Douglas W. Oard
College of Library and Information Services
University of Maryland, College Park, MD 20742
oard@glue.umd.edu

Abstract

A technique is described for aligning TREC topic descriptions that is capable of producing a small multilingual test collection which can be used for cross-language ad-hoc and routing evaluations. Methods for measuring the degree of degradation induced by the necessary approximations are described and illustrated using examples from an evaluation of two cross-language routing techniques. Although the experiments were conducted on a relatively small test collection using existing TREC relevance judgments, the results suggest that cross-language routing is practical and that the investment required to produce a cross-language test collection for the TREC multilingual track would be justified.

1 Introduction

The principal goal of the University of Maryland's participation in the Fifth Text REtrieval Conference (TREC-5) was to evaluate the performance of advanced routing techniques. We investigated two aspects of the routing problem: construction of a feature space that is independent of the language in which a document is written, and efficient construction of reduced-dimensional feature spaces on which machine learning algorithms can be effective. Our ultimate goal is to construct an adaptive multilingual routing system which can learn (from the user's responses to existing documents) to select new documents that may not be written in the same language as the existing documents. Our work on efficient construction of reduced-dimensional feature spaces revealed that the technique we applied was inappropriate for large document collections, so our official ad-hoc retrieval run was submitted using a version of SMART that was modified locally for the ISO 8859-1 character set.¹ Additional details regarding our development of monolingual routing techniques are reported in [7].²

In this paper we describe how we have used TREC document collections and relevance judgments to evaluate the performance of two techniques that we implemented for cross-language routing because we believe that those techniques may be of use to other TREC participants. The multilingual routing systems we are exploring are based on existing approaches to ad-hoc cross-language text retrieval which seek to select documents in one language based on queries expressed in another, and on existing monolingual routing research. Our contribution has been to explore how these two bodies of research can best be exploited to satisfy the unique requirements of adaptive multilingual routing. Evaluation of the resulting systems has been our greatest challenge. The test collection we have constructed can be used to compare the effect of different cross-language mapping techniques on prediction accuracy, and we have developed a methodology for measuring the degradation introduced by the unavoidable compromises that we have made when constructing the collection. As a result, we are able to qualify the broader applicability of our results and to quantify the improvement in evaluation accuracy that would result from development of a test collection tailored to the evaluation of multilingual routing systems.

*This work has been supported in part by DOD contract MDA9043C7217, ARPA and ONR contract N00014-92-J-1929, ARPA contract DACA76-92-C009, NSF award IRI-9357731, and the Logos Corporation.

¹We used ltc term weights and participated in category B.

²Current links to most of the references cited in this paper can be found online at <http://www.ee.umd.edu/medlab/filter/>

2 Adaptive Multilingual Routing

We have surveyed text routing techniques elsewhere [6], so here we describe only the technique which we have chosen to apply. Our approach is based on the ranked output paradigm in which the routing system seeks to rank order newly arrived documents with the most useful documents near the top of the list. We have based our work on a technique developed by Dumais for monolingual routing in which Latent Semantic Indexing (LSI) is used to develop relatively short feature vectors that describe the relevant training documents, and the mean of the relevant documents' feature vectors is used as the routing query [2]. LSI feature vectors describing newly arrived documents are then used to rank order the newly arrived documents in order of decreasing similarity with the routing query using the cosine similarity measure.

LSI feature vectors are constructed by counting the frequency with which each term occurs in a document and then using those values as input to a function which reduces the number of features by accounting for similarities in word usage. This function is automatically constructed using statistical techniques by examining a representative collection of text in which typical term usage variations are exhibited. We have applied this "LSI-mean" routing approach to evaluate the performance of two cross-language mapping techniques, so we have been careful to construct this mapping using the same document collection in order to assure the comparability of our results.

The cross-language mapping techniques we have evaluated were motivated by earlier work on multilingual text retrieval, a topic we have also surveyed [8]. The most obvious is to pass every document through an automatic machine translation system. In ad-hoc cross-language retrieval it is the topic specification which is most often translated. While the brevity of typical topic specifications makes that choice efficient, use of machine translation with the LSI-mean routing technique requires that every document be translated into a single language because the LSI-mean routing query is a vector made up of elements which do not correspond to individual words. Our approach, which we call "Text Translation," effectively reduces cross-language routing to its monolingual counterpart.

A second technique, "Cross-Language Latent Semantic Indexing," exploits the ability of LSI to identify and suppress the effect of word usage variations. In Cross-Language Latent Semantic Indexing, bilingual or multilingual documents are prepared by adjoining versions of the same document in different languages. LSI is then trained on that document collection to find a feature vector mapping which accepts documents from any of the languages [3, 4]. It is our interest in this technique which led us to choose the LSI-mean technique as the routing method.

Other approaches to multilingual routing are possible as well, and we have used the same methodology to evaluate a third technique which we call Vector Translation. We limit our discussion here to Text Translation and Cross-Language Latent Semantic Indexing since two techniques suffice to illustrate the way in which we have used TREC collections and relevance judgments and our work with Vector Translation is still quite preliminary.

3 Ideal Experiment Design

Routing experiments of the type we are conducting require a document collection for which relevance judgments are available, so it would be ideal if a test collection existed in which every document has versions in two languages and relevance judgments with respect to a number of standardized topics. While we ultimately intend to provide users with systems which adapt in nearly real time, for our evaluation we have chosen to introduce an artificial division between the construction of a routing query and the use of that routing query to rank order documents. We could achieve this by dividing an ideal test collection into two partitions, one for query construction and one for effectiveness evaluation. Because we wish to measure the effectiveness of cross-language selection, we use the documents in English from one partition and their associated relevance judgments to develop the routing query. We then apply a cross-language ranking system to rank order the Spanish documents from the other partition, using their associated relevance judgments to determine the quality of that ranking. We have chosen English for query construction and Spanish for evaluation because the Logos machine translation system we used for the Text Translation experiments was capable of unidirectional English to Spanish translation. We used the same selections for the Cross-Language Latent Semantic Indexing experiment in order to obtain comparable results.

Partition	English	Spanish	Relevance Judgments
Cross-Language Training	X	X	
Query Construction	X		X
Effectiveness Evaluation		X	X

Table 1: Ideal multilingual routing test collection.

Source	English	Spanish	English Rel.	Spanish Rel.
1990-1992 UN Documents	X	X		
1990-1992 Wall St Journal	X		X	
1992 El Norte Newspaper		X		X

Table 2: Evaluation using existing collections.

In Cross-Language Latent Semantic Indexing we seek to extract statistical information about word cooccurrence from a large collection of documents in which every document is duplicated in each language. In order to apply that technique we need to select a third partition of the test collection from which we can extract collocation information. It would not be reasonable to reuse one of the existing partitions for this “cross-language training” task because cross-language techniques would not be needed if all of the documents in either the query construction or the evaluation partition were already available in both languages. Relevance judgments are not needed for this language training partition. Table 1 shows which parts of the three partitions of an ideal test collection are needed.

4 Use of TREC Collections

We are aware of no large collection of the type shown in Figure 1 (but for a description of a smaller collection in Korean and English see [5]), and large collections are needed for evaluation of adaptive routing systems. Large bilingual and trilingual document collections do exist, but construction of the required topics and relevance judgments would have been well beyond our resources. TREC has provided large monolingual collections with associated topics and relevance judgments, but translation of each document into a second language would have been even more difficult. Because none of the three partitions shown in Table 1 must be both bilingual and scored, it is possible to use three existing collections to approximate the results that would be achieved using an ideal test collection. The collections we have chosen are shown in Table 2. The UN collection is a large collection of United Nations documents, each of which is available in either two or three languages (English, Spanish and French) from the Linguistic Data Consortium³. This is the same collection that was used for cross-language ad-hoc retrieval experiments by the New Mexico State University team in TREC-4 [1]. The topics and relevance judgments for the Wall Street Journal and El Norte collections are obtained from TIPSTER disk 2 and from the TREC-4 multilingual track respectively. We chose the Wall Street Journal collection on disk 2 because relevance judgments are available for that collection on all 300 topics and because that collection contains material from the same time period as the El Norte collection. For consistency, we used only those UN documents that were prepared during the same years as the Wall Street Journal articles that we used.

We performed topic alignment manually, examining each of the 50 Spanish topics and then scanning a list of the 300 available English topics in order to identify possible matches. The detailed topic descriptions were then compared and a set of topic pairs which appeared to be closely aligned were selected. Table 3 shows the five Spanish topics for which we have found closely corresponding English topics.⁴ Although the

³Information on the availability of the UN collection can be obtained from <http://www ldc.upenn.edu>

⁴Some more weakly aligned topic pairs that might also be useful are identified in [9].

Abbreviated Spanish Language Topic		Abbreviated English Language Topic	
SP10	Mexican Narcotic Trafficking	284	International Drug Enforcement
SP18	Foreign Car Makers in Mexico	290	Foreign Car Makers in the U.S.
SP22	Mexican Inflation	008	Economic Projections
SP25	Mexican Privatization Programs	128	Privatization of State Assets
SP47	Mexican Cancer Cause Research	123	Carcinogen Research and Control

Table 3: Closely related English and Spanish TREC topics.

topic descriptions in each pair have some differences, there is sufficient apparent overlap to suggest that a minimal adjustment to the sets of relevant documents would result in comparable sets of documents in the two languages. In fact, our experimental results confirm that it is possible to use the relevance judgments without any adjustment when the goal is to compare different cross-language mapping techniques.

Two potential problems arise when the three existing collections in Table 2 are substituted for the single collection shown in Table 1. The first is that the subjects addressed by the UN, the Wall Street Journal and El Norte would be expected to differ significantly. We refer to this problem as a “domain shift,” between the collections since it is caused by differences in the topical domains of the two collections. A potentially even more serious problem is that the Wall Street Journal and El Norte articles were judged against topics which have been aligned after the fact, and that alignment is far from perfect. We call this problem “topic shift.”

The domain shift between the UN documents and El Norte is fairly easy to evaluate. In order to ensure that we obtain comparable results, we have chosen to use the LSI-mean routing technique for Text Translation and Cross-Language Latent Semantic Indexing. Since Text Translation produces Spanish documents as an intermediate step, we can measure the effect of the domain shift by running the Text Translation experiment a second time. In that second run we substitute the El Norte documents for the Spanish UN documents when generating the mapping that produces the LSI feature vectors. The resulting LSI mapping will be better suited to the El Norte articles, and the difference in our precision measure reveals the effect of the domain shift between the UN collection and the El Norte collection. We have not developed any similar technique to reveal the effect of the topic shift between either of those collections and the Wall Street Journal collection.

We can estimate the effect of the topic shift by comparing cross-language and within-language performance. This could be done by dividing the El Norte collection into two partitions and then performing a monolingual evaluation in which one partition is used for query construction and the other for evaluation. That would remove the effect of the topic shift completely, although it would simultaneously remove the effect of errors introduced by the cross-language mapping technique. The effect of translation errors on the performance of the Text Translation technique are easily measured, however, using a modification of the basic Cross-Language Latent Semantic Indexing experiment. With Cross-Language Latent Semantic Indexing, LSI feature vectors can be produced from either English or Spanish documents. If the English Wall Street Journal articles are translated into Spanish before being used for query construction in the Cross-Language Latent Semantic Indexing experiment, the observed reduction in precision will be entirely attributable to errors introduced by the machine translation step. These are exactly the same errors that affect the Text Translation experiment, so this result will reveal the necessary adjustment to the difference between the monolingual evaluation on El Norte and the standard Text Translation experiment. In our initial experiments we have used the entire El Norte collection for both training and evaluation when evaluating the topic shift. Those results overstate the effect of the topic shift because they evaluate memory, not prediction accuracy, but they do provide an upper bound on the magnitude of the topic shift, and that upper bound proved to be adequate to recognize one case in which an extreme topic shift made an apparently well-aligned topic pair unusable.

Topic Pair	Technique		
	CL-LSI	TT	None
SP22/008	0.17	0.17	0.06
SP25/128	0.08	0.10	0.03
SP47/123	0.07	0.06	0.00

Table 4: Multilingual routing experiment results (precision at 0.1 recall).

5 Results

Our primary objective is to determine the *relative* performance of two cross-language routing techniques. We would expect to find the largest absolute differences in precision near the top of the ranked list, and hence we felt that vales of precision at low recall would best reveal differences in performance between the two cross-language routing techniques that we tried. Thus, rather than report average precision, we have chosen to report precision only at a fixed value of recall (0.1—the point at which 10% of the relevant documents have been seen.) The density of relevant documents is greatest near the top of the ranked list, so differences in cross-language mapping effectiveness should be most apparent at in that region. In our experiments, a recall of 0.1 is achieved after 35, 36 or 8 documents (for topics SP22, SP25 and SP47 respectively) have been found. Since that should be an adequate number of relevant documents for many types of interactive applications, the precision values we report should be representative of what might be experienced by interactive users.

We used the SMART text retrieval system, modified locally to include the LSI-mean routing technique, with ltc term weights for our experiments. We substituted morphological roots provided by the Rank Xerox morphological tagger for SMART stemming because a third technique that we are developing (Vector Translation) could potentially benefit from compatibility with a bilingual dictionary. Relevance judgments for topics 284 and 290 were not available when we ran our experiments, so we were only able to use the last three topic pairs that are shown in Table 3. Table 4 shows results for the two cross-language routing techniques, Cross-Language Latent Semantic Indexing (CL-LSI) and Text Translation (TT), and a baseline run (labeled “None”) in which we used no cross-language mapping technique at all. These results are described in detail in [9]. In this paper we will limit our comments to those which address fundamental evaluation issues.

The most significant observation that we drew from our experiments is that multilingual routing appears to be practical and that the corpora we used are adequate to demonstrate that. Both corpus-based techniques (such as Cross-Language Latent Semantic Indexing) and knowledge-based techniques (such as Text Translation) have demonstrated better performance than that which could be achieved with no translation component, despite the limitations imposed by the topic and domain shifts. This fact should also be of interest to researchers working on corpus-based ad-hoc cross-language retrieval, since it confirms that (for these three topics, at least), the UN collection and the El Norte collection are sufficiently similar to produce much better precision near the top of the ranked list than that which could be achieved by random selection. In every case the precision achieved by random selection would have been below 0.01 at any value of recall. Additional details on this point are presented in [9].

Another interesting observation is that the results without cross-language mapping exhibit a surprising amount of variation. We attribute this effect to the existence of words which are common to Spanish and English that are useful for recognizing documents that are relevant to some topics. This observation has led us to conclude that when the available corpora limit a cross-language routing or retrieval experiment to a small number of topics, a baseline run with no cross-language mapping is a simple way to gain some useful insight into the significance of the results.

Table 5 shows the results of the domain shift experiment. In two cases out of three, the domain shift between the UN collection and the El Norte collection appears to be substantial but not overwhelming. The lack of a clear domain shift effect in the third case is at least partially explained by poor performance of the LSI-mean routing technique on topic SP25. In a completely monolingual evaluation of “memory” (LSI training, query construction and evaluation all using the complete El Norte collection), the precision

Topic Pair	LSI training	
	Spanish UN	El Norte
SP22/008	0.17	0.28
SP25/128	0.10	0.10
SP47/123	0.06	0.17

Table 5: Domain shift results for Text Translation (precision at 0.1 recall).

Topic Pair	Experiment Design		CL-LSI Query Construction	
	Multilingual	Monolingual	English WSJ	Translated WSJ
SP10/022	0.02	0.20	0.01	0.01
SP22/008	0.17	0.46	0.17	0.14
SP25/128	0.10	0.10	0.08	0.13
SP47/123	0.06	0.45	0.07	0.02

Table 6: Preliminary topic shift results (precision at 0.1 recall).

achieved by the LSI-mean technique at 0.1 recall was only 0.18. This poor performance could result from a number of factors (e.g., we used less than 2% of the available documents when El Norte was used for LSI training and those documents may have been poorly chosen), and we have not yet completed our evaluation of the cause of this deficiency.

Table 6 shows preliminary results which provide bounds on the magnitude of the topic shift effect. Results for a fourth topic pair which we tried, SP10/022, are shown as well in order to illustrate the topic shift effect clearly. It appeared from inspection of the topic descriptions that topics SP10 and 022 were as similar as any of the other pairs we had chosen, but these results clearly reveal that SP10/022 is not a useful topic pair. Again, the SP25/128 topic pair yields unusual and as yet unexplained results, actually increasing precision when translation errors are introduced. The remaining two topic pairs show relatively large topic shift effects (although these are only upper bounds) after considering the relatively small translation error effects.

6 Future Work

Although we are able to estimate (or at least bound) the effect of the topic shift, it would clearly be better if a test collection were available with relevance judgments for documents in several languages with respect to an identical set of topics. The TREC-6 cross-language track will provide an initial step in this direction, assembling a document collection that could be used in future years for such an evaluation. One possible approach to exploiting this collection would be to design English topics in the main ad hoc retrieval task for which a suitable number of relevant documents would also be found in the cross-language collections. Relevance judgments generated in the ad hoc task could then be used in future years to build routing queries for the cross-language track. Although this would make topic selection for the ad hoc task more complex, it has the advantage of creating a one-direction (English to another language) cross-language routing evaluation collection without the expense of implementing pooled relevance assessments for the full cross-language track.

7 Conclusions

We have developed a way to apply existing TREC collections to compare the effectiveness of cross-language mapping techniques in an adaptive multilingual routing system. The domain shift effect we have described will be inherent in corpus-based techniques such as Cross-Language Latent Semantic Indexing, unless collec-

tions of translated texts which use language in almost exactly the same way as the training and evaluation documents can be found. Thus, the ability to characterize the magnitude of the domain shift effect will be important whenever knowledge-based and corpus-based techniques are compared. The topic shift effect, on the other hand, is strictly an artifact of our experiment design. It is not possible to draw broadly applicable conclusions from only three topic pairs, but our results do at least indicate that the additional investment required to produce a cross-language test collection would be well justified because evaluation of adaptive multilingual routing techniques appears to be both practical and productive.

Acknowledgments

The author would like to express his appreciation to Bonnie Dorr and Michael Littman for their comments on earlier draft of this work, to David Hull of Rank Xerox Research Centre for his assistance with data preparation, and to the Logos corporation for machine translation support.

References

- [1] Mark Davis and Ted Dunning. A TREC evaluation of query translation methods for multi-lingual text retrieval. In D. K. Harman, editor, *The Fourth Text Retrieval Conference (TREC-4)*. NIST, November 1995.
- [2] S. T. Dumais. Latent Semantic Indexing (LSI): TREC-3 report. In Donna Harman, editor, *Overview of the Third Text REtrieval Conference*, pages 219–230. NIST, November 1994.
- [3] Susan T. Dumais, Thomas K. Landauer, and Michael L. Littman. Automatic cross-linguistic information retrieval using latent semantic indexing. In Gregory Grefenstette, editor, *Working Notes of the Workshop on Cross-Linguistic Information Retrieval*. ACM SIGIR, August 1996.
- [4] Thomas K. Landauer and Michael L. Littman. Fully automatic cross-language document retrieval using latent semantic indexing. In *Proceedings of the Sixth Annual Conference of the UW Centre for the New Oxford English Dictionary and Text Research*, pages 31–38. UW Centre for the New OED and Text Research, Waterloo Ontario, October 1990.
- [5] Joon Ho Lee and Jeong Soo Ahn. Using n-grams for Korean text retrieval. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, August 1996.
- [6] Douglas W. Oard. The state of the art in text filtering. *User Modeling and User-Adapted Interaction*, 1997. To appear.
- [7] Douglas W. Oard and Nicholas DeClaris. Cognitive models for text filtering. Technical Report EE-TR-96-28, University of Maryland, College Park, 1996.
- [8] Douglas W. Oard and Bonnie J. Dorr. A survey of multilingual text retrieval. Technical Report UMIACS-TR-96-19, University of Maryland, Institute for Advanced Computer Studies, April 1996.
- [9] Douglas William Oard. *Adaptive Vector Space Text Filtering for Monolingual and Cross-Language Applications*. PhD thesis, University of Maryland, College Park, August 1996.



An Investigation of Relevance Feedback Using Adaptive Linear and Probabilistic Models

Robert G. Sumner, Jr., and W. M. Shaw, Jr.
School of Information and Library Science
University of North Carolina
Chapel Hill, NC 27599-3360 USA
{sumnr, shaw}@ils.unc.edu

Abstract

The SMART system (v. 11.0) was used as a front-end to a two-stage retrieval process. In the first stage, (WSJ) documents and the description field of (ad hoc) topics were indexed by the stems of single terms; *lnc* and *ltc* weights were computed for word stems in documents and queries, respectively; and documents were ranked according to the cosine similarity of document and query vectors. Related by the initial query vector, the first 5000 documents in the ranked list for each topic constituted a "condensed" database for that topic. Preliminary experiments with TREC-4 topics and official relevance evaluations suggested each such database would include a high fraction of relevant documents for the associated topic, and the result was confirmed by TREC-5 results. In the second stage, initial query vectors were automatically refined by two relevance feedback strategies applied to the condensed databases. One of us employed the adaptive linear model (*uncis1*), and the other used a variation of the "classic" probabilistic model (*uncis2*); relevance judgments were made independently. In *uncis1*, the query at a given search iteration is expanded by all terms in relevant, retrieved documents and all terms in selected, nonrelevant, retrieved documents, and documents are ranked by the inner product of document and query vectors. In *uncis2*, the query is expanded by all terms in relevant, retrieved documents, and documents are ranked by the cosine similarity of document and query vectors. For *uncis1* and *uncis2*, respectively, average non-interpolated precision values over all relevant documents are 0.25 and 0.20, and average R-precision values are 0.25 and 0.21. Results show that independent relevance judgments made in *uncis1* and *uncis2* are quite different and have a strong effect on retrieval outcomes; our relevance evaluations also differ significantly from official relevance judgments. Retrieval performance improves when official relevance judgments are utilized by both models. For the 31 topics in which there was an official relevant document in the top 34 of the initial ranking, average non-interpolated precision values are 0.60 for the adaptive linear model and 0.59 for the probabilistic model.

Introduction

The next generation of information retrieval (IR) systems can be expected to be interactive, adaptive, and intelligent. Such systems will allow end-user relevance assessments and end-user query modifications to guide the construction of effective representations of information need. Incorporation of relevance feedback in IR experimentation has been found to improve retrieval performance as shown, for example, by Buckley, Singhal, Mitra, and Salton (1996); Haines and

Croft (1993); Robertson, Walker, Beaulieu, Gatford, and Payne (1996); and Salton and Buckley (1990). Among numerous unresolved issues related to end-user and IR system interactions, we addressed two questions in our experiments for TREC-5.

- (1) Is the effectiveness of relevance feedback adversely affected by concentrating on a small subset of a document collection defined by an initial query statement?
- (2) What is the relative effectiveness of alternative relevance feedback models in the context of end-user relevance evaluations and automatic query expansion?

Two-Stage Process

Information retrieval has been viewed as a two-stage process (Boyce, 1982; Shalini, 1993). In the context of the present investigation, an initial ranking of documents is produced for a topic and constitutes the outcome of the first stage. In the second stage, the top n documents of the initial ranking serve as a "condensed" collection for further retrieval processing. For example, Buckley, Salton, Allan, and Singhal (1995) selected the first 1750 documents from a "global" search of the TREC-3 database, reranked the 1750 documents according to "local" criteria, and found that the resulting ad hoc run (*CrmLA*) compared favorably with a run applied to the complete database (*CrmLEA*). If the top n documents related by an initial query vector include a high fraction of the documents relevant to that query, applying relevance feedback to the condensed collection offers an efficient strategy for producing a satisfactory retrieval outcome. If the top n documents do not include a high fraction of relevant documents and the retrieval outcome is not satisfactory, the resulting query vector, refined by several iterations of the feedback process, can be resubmitted to the full database for further processing. In our TREC-5 runs, the second stage of the retrieval process focuses on the use of relevance feedback strategies. We are testing the effectiveness of relevance feedback algorithms applied to a condensed collection for each topic; we have not yet explored the utility of submitting the refined query vector to the entire database.

The effectiveness of our feedback runs is influenced by the proportion of relevant documents among the top n documents of the initial ranking; if a relevant document is not in the top n , then it cannot be retrieved. To determine an appropriate value for n , we conducted preliminary experiments using TREC-4 ad hoc topics and official relevance assessments for the portion of the Wall Street Journal (WSJ) collection used in category B runs. An initial ranking of documents was created for each topic using the same ranking procedure that was employed in our TREC-5 runs. The proportion of relevant documents that are in the top n documents (i.e., recall) was determined for several values of n . For a given value of n , Table 1 shows the number of topics with a recall value within the stated range. The 45 topics evaluated are those with at least one relevant document in the collection. As shown in Table 1, 100% of the relevant documents appear in the first 5000 documents for 30 of 45 topics, and 80% or more of the relevant documents appear in the first 5000 documents for 43 of the 45 topics.

Table 1: For a given value of n , the number of TREC-4 ad hoc topics with the value for recall within the stated range.

Recall (Rec)	Value for n				
	2000	3000	4000	5000	6000
$Rec = 1.000$	21	26	28	30	30
$0.900 \leq Rec < 1.000$	8	7	7	7	7
$0.800 \leq Rec < 0.900$	7	6	6	6	6
$0.700 \leq Rec < 0.800$	4	4	2	0	0
$0.600 \leq Rec < 0.700$	2	0	0	0	0
$0.500 \leq Rec < 0.600$	2	1	1	1	1
$0.400 \leq Rec < 0.500$	0	0	1	1	1
$0.300 \leq Rec < 0.400$	1	1	0	0	0

Initial Ranking of Documents

SMART (v. 11.0) was used to index the WSJ collection and the fifty ad hoc topics. For each document, <HL>, <LP>, and <TEXT> fields were indexed. These three fields correspond to the headline, lead paragraph, and remaining paragraphs, respectively, of a WSJ article. Only the <desc> field of the topics was indexed. SMART indexed the documents and topics by removing stop words and stemming the remaining terms with the modified Lovins algorithm.

In SMART, each document i is represented by a document vector \mathbf{d}_i , and the query is represented by a query vector \mathbf{q} . Each component d_{ik} of \mathbf{d}_i is the weight of term k in document i . Likewise, each component q_k of \mathbf{q} is the weight of term k in the query. The document and query vectors reside in a t -dimensional vector space, where t is the number of terms in the indexing vocabulary.

Only single terms were used in the present investigation. Document term weights were SMART *lnc* weights, and query term weights were SMART *ltc* weights (Buckley et al., 1995). The formula for the *lnc* document term weight is

$$d_{ik} = \frac{\log(f_{ik}) + 1}{\sqrt{\sum_{j=1}^t (\log(f_{ij}) + 1)^2}}, \quad (1)$$

where f_{ik} is the number of times term k appears in document i . The within-document frequency, f_{ik} , is an indication of the degree to which document i is about the concept represented by term k ; thus, as f_{ik} increases, d_{ik} increases. The denominator of Equation 1 is a document-length normalization factor; this normalization keeps short documents from being penalized.

The formula for the *ltc* query term weight is similar to *lnc*:

$$q_k = \frac{(\log(f_k) + 1) * idf_k}{\sqrt{\sum_{j=1}^t [(\log(f_j) + 1) * idf_j]^2}} \quad (2)$$

where f_k is the number of times term k appears in the query, and idf_k is the inverse document frequency (Sparck Jones, 1972) of term k . Like within-document frequency, the within-query frequency, f_k , is an indication of the degree to which the query is about the concept represented by term k ; thus, as f_k increases, q_k increases. As in Equation 1, the denominator of Equation 2 is a length normalization factor. Finally, the inverse document frequency, idf_k , is computed by $\log(N/n_k)$ where N is the number of documents in the collection and n_k is the number of documents in the collection that are indexed by term k . The greater the number of documents an index term k represents, the lower the power of k to discriminate documents. Thus, as n_k increases, idf_k decreases and q_k decreases.

SMART created the initial ranking of documents. The documents were ranked in decreasing order of the dot-product $\mathbf{q} \bullet \mathbf{d}_i$, where

$$\mathbf{q} \bullet \mathbf{d}_i = \sum_{k=1}^t q_k d_{ik} \quad (3)$$

The dot product of \mathbf{d}_i and \mathbf{q} , whose components are defined by Equations 1 and 2, respectively, is the cosine similarity of the two vectors (Salton & McGill, 1983, p. 121).

Relevance Feedback Algorithms

Two relevance feedback models were studied. Run *uncis1* employed the adaptive linear (AL) model (Wong & Yao, 1990). Run *uncis2* employed a probabilistic model, following Robertson and Sparck Jones (1976).

In an effort to produce comparable results, several aspects of the runs were controlled. First, as discussed previously, the top 5000 documents of the SMART initial ranking for a topic served as a condensed collection for relevance feedback operations on that topic. Second, relevance feedback operations ranked documents in decreasing order of $\mathbf{q}_m \bullet \mathbf{d}_i$, where \mathbf{q}_m is the query vector for iteration m . Here, the initial query vector is \mathbf{q}_1 , the modified query vector after the first set of relevance evaluations is \mathbf{q}_2 , and so on. Defined by Equation 1, components of the document vector \mathbf{d}_i were held constant for all iterations of both runs. Thus, the advantages of employing within-document frequency and document-length normalization apply to the relevance feedback iterations as well as to the initial ranking. Third, a query was expanded by including all terms in retrieved, relevant documents. (For some of the *uncis1* iterations, all terms in selected nonrelevant documents were included as well.) Some justification for expanding the query by all of the terms in retrieved, relevant documents is provided by Salton and Buckley (1990). They examined a number of different relevance feedback models and several different expansion methods, and found that expanding the query by all terms in relevant documents generally produced better results than other expansion methods studied. However, in contrast to the WSJ collection, the collections studied in Salton and Buckley's experiment consisted of document surrogates (such as abstracts and titles). Haines and Croft (1993), studying the full-text collection, WEST, found that for some expansion methods, adding too many terms to the query caused a substantial decrease in performance. They did not find such a decrease in the document-surrogate collection, CACM. Fourth, query term weights were cumulated as follows:

$$\mathbf{q}_{m+1} = \mathbf{q}_m + \mathbf{x}. \quad (4)$$

The query vector \mathbf{q}_{m+1} for the iteration $m+1$, is created by the vector sum of the query vector \mathbf{q}_m for the current iteration m and the vector \mathbf{x} . The vector \mathbf{x} depends on the algorithm employed by each run. For example, for many of the *uncis1* iterations, \mathbf{x} was the vector sum of all of the *new* relevant documents retrieved by \mathbf{q}_m . (Here, a “new” relevant document is one that was not retrieved in a prior iteration.) An advantage of cumulating query term weights is that information from previous iterations contributes to the new query vector. Thus, the query vector does not veer off radically in a new direction due to the influence of new documents retrieved in the latest iteration. Finally, the official run submitted for a topic was the SMART initial ranking if no documents retrieved from this ranking were declared relevant. Otherwise, the ranking resulting from the final set of relevance evaluations was submitted for a topic.

We did not control for the number of iterations or the number of documents assessed in each iteration. All top-ranked documents retrieved in an iteration were assessed for relevance. An iteration (and a search) generally ended when the searcher felt there would be no further benefit to continue to assess documents. Relevance judgments were made independently in *uncis1* and *uncis2* and appear to have undermined all other efforts to produce comparable results; that is, to produce comparisons attributable primarily to the algorithms.

adaptive linear model (run *uncis1*)

Run *uncis1* employed the adaptive linear (AL) model (Bollmann & Wong, 1987; Wong & Yao, 1990; Wong, Yao, & Bollmann, 1988; Wong, Yao, Salton, & Buckley, 1991). The AL model is based on the concept of the preference relation (Fishburn, 1970; Roberts, 1979). The *user preference relation* \succ on a set of documents D is defined as a binary relation on D where for all $d, d' \in D$,

$$d \succ d' \Leftrightarrow \text{the user with a query prefers } d \text{ to } d' \quad (5)$$

(Bollmann & Wong, 1987). An IR model based on the user preference relation allows the use of a multivalued relevance scale—for example, “relevant,” “marginally relevant,” and “not relevant.” However, for TREC-5, only binary relevance distinctions were made.

Wong et al. (1991) assume that \succ on D can be mapped to a corresponding relation on \mathbf{D} , the set of document vectors for the documents in D . Thus, for all $d, d' \in D$,

$$d \succ d' \Leftrightarrow \mathbf{d} \succ \mathbf{d}', \quad (6)$$

where d and d' are represented, respectively, by \mathbf{d} and \mathbf{d}' and where \succ is also used for the relation on \mathbf{D} for the sake of convenience. The relation \succ on \mathbf{D} is defined to be *weakly linear* if there exists a query vector \mathbf{q} such that for all $\mathbf{d}, \mathbf{d}' \in \mathbf{D}$,

$$\mathbf{d} \succ \mathbf{d}' \Rightarrow \mathbf{q} \cdot \mathbf{d} > \mathbf{q} \cdot \mathbf{d}' \quad (7)$$

(Wong & Yao, 1990; Wong et al., 1988). Such a query vector is called a *solution vector*. In the context of binary relevance judgments, if \succ on \mathbf{D} is weakly linear, then a solution vector \mathbf{q} will rank all of the relevant documents in the collection before all of the nonrelevant ones.

If \succ on \mathbf{D} is weakly linear, then a solution vector can be found by employing an *error-correction procedure* (Nilsson, 1965, Ch. 4; Wong et al., 1988). However, a user's preferences are only known for the set of documents (the *sample set*) that have been retrieved and assessed for relevance; a document in the sample set may have been retrieved during previous search iterations or the current one. Thus, in the AL model, a solution vector for \mathbf{S} is found, where \mathbf{S} is the set of document vectors corresponding to the sample set S . Wong and Yao (1990) assume that, as \mathbf{S} grows larger, the solution vector for \mathbf{S} approaches the solution vector for \mathbf{D} . A solution vector for \mathbf{S} can almost always be found; \succ on a set of document vectors is usually weakly linear when the number of terms in the indexing vocabulary is much greater than the number of vectors in the set (Nilsson, pp. 32-35).

For TREC-5, the solution vector for \mathbf{S} was determined by employing the error-correction procedure used by Wong et al. (1991). If \succ on \mathbf{S} is weakly linear, it can be shown that this algorithm will converge to a solution vector (Duda & Hart, 1973, pp. 142-145; Nilsson, 1965, pp. 85-87). The error-correction procedure is an iterative algorithm. During a given cycle i of the algorithm, if query vector $\mathbf{q}_{(i)}$ is a solution vector, then the algorithm terminates. If $\mathbf{q}_{(i)}$ is not a solution vector, a new query vector $\mathbf{q}_{(i+1)}$ is created by

$$\mathbf{q}_{(i+1)} = \mathbf{q}_{(i)} + \mathbf{b}_{\max} \quad (8)$$

where

$$\mathbf{B} = \{\mathbf{b} = \mathbf{d} - \mathbf{d}' \mid \mathbf{d}, \mathbf{d}' \in \mathbf{S} \text{ and } \mathbf{d} \succ \mathbf{d}'\} \quad (9)$$

and where $\mathbf{b}_{\max} \in \mathbf{B}$ such that for all $\mathbf{b} \in \mathbf{B}$,

$$-\mathbf{q}_{(i)} \bullet \mathbf{b}_{\max} \geq -\mathbf{q}_{(i)} \bullet \mathbf{b}. \quad (10)$$

As the desired result is $\mathbf{q}_{(i)} \bullet \mathbf{b} > 0$ for all $\mathbf{b} \in \mathbf{B}$, then the quantity $-\mathbf{q}_{(i)} \bullet \mathbf{b}$ can be viewed as a measure of the extent to which \mathbf{b} is in error; \mathbf{b}_{\max} then is that \mathbf{b} that produces the maximum error (Wong et al., 1991). In TREC-5, only binary relevance assessments were made. Accordingly, in the context of binary relevance judgments,

$$\mathbf{q}_{(i+1)} = \mathbf{q}_{(i)} + \mathbf{d}_{\text{worst_rel}} - \mathbf{d}_{\text{best_nonrel}} \quad (11)$$

where the documents in S have been ranked by $\mathbf{q}_{(i)}$, $\mathbf{d}_{\text{worst_rel}}$ represents the worst-ranked relevant document, and $\mathbf{d}_{\text{best_nonrel}}$ represents the best-ranked nonrelevant document.

The *starting vector* $\mathbf{q}_{(0)}$ is the initial query vector of the error-correction procedure (Wong et al., 1991). (If the starting vector is a solution vector, then the procedure terminates without creating $\mathbf{q}_{(1)}$.) Wong et al. conducted relevance feedback runs using three different starting vectors, and the best results were for those runs where the starting vector was the user's original query vector plus the vectors for the retrieved, relevant documents. Wong et al. conducted only

one feedback iteration; because we conducted more than one feedback iteration, our starting vector consisted of the query vector for the current search iteration plus the vectors for all *new* relevant documents retrieved.

Using binary relevance assessments, Wong et al. (1991) compared the AL model to some other relevance feedback models that had been found to produce good results. The AL model's average precision was slightly superior to that of these other models. The five test collections studied in the Wong et al. experiment consisted of document surrogates (such as abstracts and titles) instead of full-text documents. Therefore, we felt that it would be interesting to evaluate the AL model using the full-text and much larger WSJ collection.

probabilistic model (run *uncis2*)

As in the implementation of the AL model, the query vector at search iteration $m+1$, \mathbf{q}_{m+1} , is the vector sum of the query vector at iteration m , \mathbf{q}_m , and the vector \mathbf{x} . The complete set of documents evaluated by the searcher in the ranked outcome of iteration m defines the *training set*, and \mathbf{x} includes all terms in relevant documents of this set. The weight of term k in \mathbf{x} is denoted by $(tr)_k$ and is defined by

$$(tr)_k = \log \left[\frac{p_k / (1 - p_k)}{u_k / (1 - u_k)} \right], \quad (12)$$

where p_k is the probability term k appears in a relevant document of the training set, and u_k is the probability term k appears in a nonrelevant document in the training set (Robertson & Sparck Jones, 1976). Definitions of p_k and u_k can lead to undefined values of $(tr)_k$, when a term appears in all or none of the relevant or nonrelevant documents in the training set, and consequently, computing equations must be used to estimate the probabilities. Because the "conventional" 0.5 formula (Robertson & Sparck Jones) can overestimate term relevance weights, especially when few relevant documents are detected in the training set (Shaw, 1995; van Rijsbergen, Harper, & Porter, 1981; Yu, Buckley, Lam, & Salton, 1983), "alternative" computing equations were used to determine p_k and u_k (Shaw, 1995):

$$p_k = \frac{r_k}{N_r} \left[\begin{array}{l} \frac{1}{N_d^2}, \text{ if } r_k = 0 \\ 1 - \frac{1}{N_d^2}, \text{ if } r_k = N_r \end{array} \right] \quad (13)$$

and

$$u_k = \frac{d_k - r_k}{N_d - N_r} \left[\begin{array}{l} \frac{1}{N_d^2}, \text{ if } d_k - r_k = 0 \\ 1 - \frac{1}{N_d^2}, \text{ if } d_k - r_k = N_d - N_r \end{array} \right], \quad (14)$$

where N_d and N_r are the total number of documents and the total number of relevant documents, respectively, in the training set, and d_k and r_k are the number of documents and the number of relevant documents, respectively, in which term k appears. The alternative computing equations contribute to a high level of retrieval effectiveness in retrospective and predictive tests in a small retrieval test collection (Shaw, 1995, 1996) and are employed here for comparative purposes. In all feedback iterations, documents were ranked according to the cosine similarity of document and query vectors.

Results

The number of feedback iterations for *uncis1* and *uncis2* varied from 0 to 4 and from 0 to 5 respectively. If the searcher declared no document to be relevant in the SMART initial ranking, the number of feedback iterations was 0; for both *uncis1* and *uncis2*, 14 topics out of 50 had no feedback iterations. Considering only those topics with feedback iterations, the average number per topic for *uncis1* and *uncis2* were 2.78 and 2.83 respectively.

For those topics for which the searcher declared no document to be relevant in the SMART initial ranking, the average number of documents evaluated from this ranking for *uncis1* was 68.29. For all other topics, the average number of documents evaluated from the initial ranking was 26.53. The average number of *new* documents evaluated from the first feedback ranking for *uncis1* was 14.47, and this average for subsequent rankings was 8.50. Corresponding data are not recoverable for *uncis2*; however, an attempt was made to examine 30 new documents per iteration. For both *uncis1* and *uncis2*, the actual number of new documents evaluated varied widely.

We have assumed that a high fraction of relevant documents appears among the first 5000 documents ranked by SMART. A comparison of results for TREC-4 and -5 is given in Table 2 and shows that initial rankings in TREC-5 are less reliable than initial rankings in TREC-4, but are quite satisfactory. For 51% of topics with at least one relevant document in TREC-5, 100% of the relevant documents appear among the first 5000 documents, while 67% of topics in TREC-4 meet that standard. Similarly, 73% of TREC-5 topics yield 80% or more of the relevant documents in the initial rankings, while 96% of TREC-4 topics are so productive. Our results in TREC-5 are constrained by a few topics for which high recall is not possible.

Summary statistics for our TREC-5 results are given in Tables 3 and 4. These results are based on the 45 topics with at least one relevant document. Results for a baseline run consisting of the SMART initial ranking are also given.

Table 2: For $n = 5000$, the number of TREC-4 and TREC-5 ad hoc topics with the value for recall within the stated range.

Recall (<i>Rec</i>)	TREC-4 Topics	TREC-5 Topics
$Rec = 1.000$	30	23
$0.900 \leq Rec < 1.000$	7	7
$0.800 \leq Rec < 0.900$	6	3
$0.700 \leq Rec < 0.800$	0	5
$0.600 \leq Rec < 0.700$	0	2
$0.500 \leq Rec < 0.600$	1	2
$0.400 \leq Rec < 0.500$	1	1
$0.300 \leq Rec < 0.400$	0	1
$0.200 \leq Rec < 0.300$	0	0
$0.100 \leq Rec < 0.200$	0	0
$0.000 < Rec < 0.100$	0	0
$Rec = 0.000$	0	1

Table 3: Recall level precision averages and average non-interpolated precision for baseline run, *uncis1*, and *uncis2*.

Recall Level Precision Averages			
Recall	baseline precision	<i>uncis1</i> precision	<i>uncis2</i> precision
0.00	0.3126	0.5704	0.5156
0.10	0.2589	0.5185	0.4299
0.20	0.2037	0.4403	0.3703
0.30	0.1643	0.3674	0.3270
0.40	0.1282	0.3242	0.2248
0.50	0.1146	0.2904	0.1957
0.60	0.0816	0.1961	0.1389
0.70	0.0696	0.1092	0.0730
0.80	0.0446	0.0706	0.0371
0.90	0.0376	0.0493	0.0209
1.00	0.0318	0.0408	0.0181
Average precision over all relevant documents			
	baseline	<i>uncis1</i>	<i>uncis2</i>
non-interpolated	0.1172	0.2510	0.1948

Table 4: Document level precision averages and average R-precision for baseline run, *uncis1*, and *uncis2*.

Document Level Averages			
Cutoff	baseline precision	<i>uncis1</i> precision	<i>uncis2</i> precision
At 5 docs:	0.1556	0.3867	0.3333
At 10 docs:	0.1444	0.3289	0.2844
At 15 docs:	0.1230	0.2785	0.2281
At 20 docs:	0.1111	0.2356	0.1956
At 30 docs:	0.0926	0.1807	0.1459
At 100 docs:	0.0487	0.0824	0.0644
At 200 docs:	0.0333	0.0488	0.0407
At 500 docs:	0.0199	0.0249	0.0215
At 1000 docs:	0.0128	0.0145	0.0135
R-Precision (precision after R (number relevant for a query) documents retrieved)			
	baseline	<i>uncis1</i>	<i>uncis2</i>
Exact	0.1428	0.2470	0.2123

Discussion

Our objective was to compare the effectiveness of AL and probabilistic models in the context of relevance feedback. With respect to overall (average non-interpolated) precision, performance outcomes for *uncis1* and *uncis2* rank first and fourth, respectively, among official results reported by Category B participants. As both runs were the only Category B runs to incorporate relevance feedback utilizing searcher relevance assessments, their high performance relative to the other runs offers evidence for the effectiveness of this retrieval strategy. However, with overall precision values ranging from 0.20 to 0.25, results are poor for both models in absolute value. A query-level analysis provides some insight into low, official levels of performance.

Table 5 gives performance rankings of topics in decreasing order of average non-interpolated precision for *uncis1*, recall values at $n=5000$, rank of the first official relevant document in the initial search, overlap of judges' and searchers' relevance assessments, and average non-interpolated precision values for both runs. The table includes 45 topics with at least one official relevant document. Focusing on recall, it can be seen that a high fraction of relevant documents appears in the condensed collection for most topics. Including topic 253 (performance rank 44), for which no officially declared relevant document is found among the first 5000 documents, an average of 87% of relevant documents for a topic appeared in the associated condensed collection. In addition, measurable values of precision are reported for standard recall value 1.00 for both runs, as shown in Table 3. The condensed document collection for each topic does not explain reported performance levels.

Table 5: Factors that may influence performance results. Only the 45 topics with an “official” relevant document are shown.

Performance Rank ¹ / Topic #	Recall at $n = 5000$	Rank of first relevant document ²	<i>uncis1</i>		<i>uncis2</i>	
			Overlap ³	Average precision ⁴	Overlap ³	Average precision ⁴
1 / 276	1.000	1	0.400	0.7000	0.286	0.6667
2 / 259	1.000	3	0.636	0.6971	0.667	0.7184
3 / 274	0.912	23	1.000	0.6786	1.000	0.5035
4 / 300	0.933	11	0.818	0.6699	0.833	0.5422
5 / 288	1.000	1	0.722	0.6073	0.700	0.4940
6 / 285	1.000	1	0.938	0.6060	0.710	0.5679
7 / 287	1.000	4	0.714	0.5909	0.300	0.1952
8 / 261	0.900	1	0.714	0.5275	0.333	0.3550
9 / 272	1.000	1	0.286	0.4675	0.500	0.4903
10 / 258	1.000	31	0.500	0.4217	0.091	0.0387
11 / 289	0.988	2	0.640	0.4159	0.606	0.3223
12 / 299	1.000	4	0.800	0.4061	0.300	0.3691
13 / 280	1.000	2	0.333	0.3803	0.400	0.3559
14 / 256	1.000	2	0.444	0.3560	0.500	0.5694
15 / 257	0.800	5	1.000	0.3546	0.333	0.1133
16 / 298	1.000	23	0.714	0.3510	0.750	0.2835
17 / 252	1.000	210	0.111	0.3333	0.063	0.1000
18 / 254	0.923	1	0.539	0.3247	0.303	0.1133
19 / 273	0.796	5	0.889	0.3157	0.833	0.2321
20 / 271	0.950	4	0.471	0.2815	0.667	0.2911
21 / 293	1.000	2	1.000	0.2611	0.833	0.3300
22 / 282	0.857	5	0.400	0.2074	0.400	0.2695
23 / 255	0.906	9	0.462	0.2015	0.444	0.1092
24 / 291	0.841	3	0.765	0.2009	0.364	0.0547
25 / 284	1.000	1	0.263	0.1706	0.263	0.1003
26 / 283	1.000	9	0.105	0.1385	0.231	0.1223
27 / 265	1.000	302	0.200	0.1290	0.000	0.0256
28 / 294	0.556	6	0.500	0.1176	0.400	0.0545
29 / 290	0.676	281	0.214	0.1093	0.385	0.0878
30 / 266	1.000	10	0.200	0.1000	0.143	0.1714
31 / 268	0.700	20	0.500	0.0600	0.073	0.0111
32 / 286	0.750	85	0.111	0.0364	0.200	0.0175
33 / 251	0.635	10	None found	0.0233	None found	0.0233
34 / 292	1.000	71	0.000	0.0155	0.000	0.0033
35 / 262	1.000	85	None found	0.0118	None found	0.0118
36 / 269	0.598	34	0.031	0.0066	0.130	0.0309
37 / 264	0.724	26	None found	0.0054	None found	0.0054
38 / 295	1.000	188	None found	0.0053	None found	0.0053
39 / 277	1.000	179	None found	0.0037	None found	0.0037
40 / 260	1.000	387	None found	0.0026	None found	0.0026
41 / 275	0.333	661	None found	0.0005	None found	0.0005
42 / 297	0.731	495	None found	0.0005	None found	0.0005
43 / 270	0.471	733	None found	0.0001	0.000	0.0013
44 / 253	0.000	> 5000	None found	0.0000	None found	0.0000
45 / 267	1.000	1863	0.000	0.0000	None found	0.0000

¹Performance rank determined by average precision value for *uncis1*.

²Rank of the first “official” relevant document in the initial ranking of documents.

³Determined by $x/(x + y)$ where x is the number of documents declared relevant by both the official TREC judges and the searcher, and y is the number of documents declared relevant by the searcher but nonrelevant by the judges. The value given for “Overlap” is “None found” if the searcher declared no document to be relevant.

⁴Average non-interpolated precision over all of the relevant documents.

Rank of first relevant document provides insight into the poor performance of some topics. Table 5 shows that performance is generally poor if the rank of the first official relevant document in the SMART initial ranking is high (greater than 60 or so). For most of these topics, the searcher, as to be expected, declared no retrieved document in the initial ranking to be relevant--indicated by "None found" in the overlap columns for *uncis1* and *uncis2*. For such cases, the ineffective initial ranking becomes the official result. Special mention should be made, however, for three topics where the rank of the first relevant document is high: topics 252, 265, and 290, at performance ranks 17, 27, and 29, respectively. For these topics, nonrelevant documents were declared relevant by the searcher and associated relevant documents were eventually retrieved. Such cases suggest a need for a multivalued relevance scale, including the category "marginally relevant" for documents a searcher considers related to a topic but not an "answer" to it.

Results reported in the table also suggest that differences in performance between the two models are often related to differences in relevance assessments of searchers and judges. Consider topic 258, at performance rank 10, for example, where the overlap for *uncis1* and *uncis2* are 0.50 and 0.09, respectively, and average precision values are 0.42 and 0.04, respectively. Clearly, it would be informative to disassociate discrepancies between relevance assessments by searchers and judges from differences in system performance.

To evaluate the influence of "incorrect" relevance assessments, we selected the 31 topics for which the first official relevant document appears at rank 34 or less, used the official list of relevant documents for all evaluations, and employed an experimental design that replicates aspects of our manual procedures. When we considered at least one document to be relevant in our manual assessments of an initial ranking, we carried out two to five feedback iterations, and evaluated about 8 to 30 new documents in each iteration. In experimental analyses presented here, three feedback iterations were employed. Query expansion and calculations of term weights were based on the complete set of retrieved, evaluated documents, which increased by increments of 25 documents after the first 35 were evaluated in the initial ranking. As in official runs, query expansion strategies employed all terms in relevant, retrieved documents for both models and all terms in selected nonrelevant documents for the AL model, and calculations were based on the sample set, defined in the section describing the AL model. Unlike query vectors computed for *uncis2*, query vectors for the probabilistic model were not cumulated from one iteration to the next.

Official results for the 31 topics are indicated in Figure 1 by the bottom two curves, and experimental results are represented by the middle pair of curves. Clearly, experimental results are superior to official results. Average non-interpolated precision increases from 0.34 (*uncis1*) to 0.60 for the AL model and from 0.27 (*uncis2*) to 0.59 for the probabilistic model, when searcher relevance judgments are replaced with official relevance judgments. (Consistently large sample sets may have also contributed to the increase in performance.) The performance advantage of the AL model indicated by official performance levels is negligible when the confounding influence of inaccurate relevance assessments by searchers are removed. The high relative improvement in performance for the probabilistic model in these implementations appears to be related to two factors:

- (1) *uncis2* results were diminished by a lower overlap with official relevance assessments than *uncis1*, and

- (2) probabilistic results were enhanced by adopting the sample set instead of the training set for term relevance calculations.

Figure 1 also includes retrospective results for both models. Retrospective results are based on the 5000 documents in the condensed collection for each topic and represent high performance limits for both models. As before, both models employ all terms in relevant documents, while the AL model also uses all terms in selected nonrelevant documents to produce a solution vector. High performance limits of the AL model exceed those of the probabilistic model because the solution vector generated by the former model insures that all relevant documents are ranked ahead of any nonrelevant document in the collection. It is informative to compare recall-precision curves based on a realistically limited view of the collection and on relevance assessments of judges (the middle pair of curves) to an unrealistically complete view of the collection (the retrospective view) and official judgments (the top two curves). When our relevance assessments match the judges', the resulting recall-precision curves are closer to "perfection" than to official results, for standard recall values less than or equal to 0.4. The capacity of systems employing relevance feedback to discriminate relevant documents from nonrelevant documents may be greater than official results suggest.

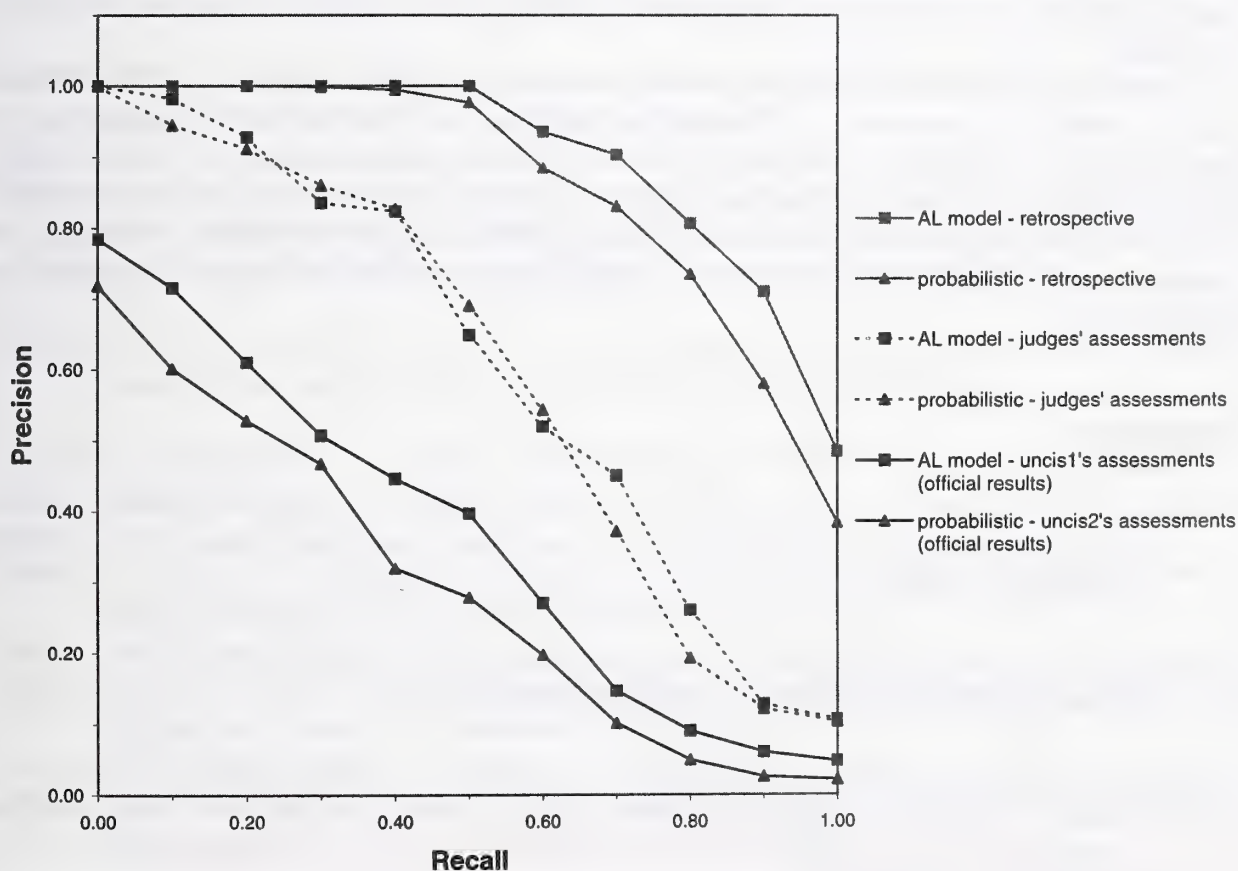


Figure 1 Recall-precision curves. Runs conducted on the 31 topics with at least one relevant document in the top 35 of the SMART initial ranking.

Conclusion

Defined by the first 5000 documents retrieved by an initial search of the WSJ database, the condensed collection for each topic provided a suitable domain for relevance feedback. The condensed collection for each topic included an average of 87% of the relevant documents and contributed to efficient execution of feedback algorithms based on the Adaptive Linear (AL) and probabilistic models. Such a process can lead to one of two outcomes: a satisfactory retrieval result, which is possible in the present context, or a refined query vector, which can be applied to the complete database. We have not yet explored the latter possibility. Implemented as an intelligent client interacting with an end-user and a file server, the two-stage process might offer an efficient and effective model for IR in large distributed databases.

Official results for the AL model are superior to those for the probabilistic model and are first among category B participants. Confounded by distinct, inaccurate relevance assessments by searchers, official results for both models are low in absolute value. Average results for both models increase significantly and become essentially equivalent, when official relevance judgments are used and other influential factors are held constant. Whether the fusion of AL and probabilistic results improve performance remains to be tested. That the AL model allows multivalued relevance assessments may cause it to be the model of choice for many applications, if such a capability is favored by end-users, which is likely, and improves performance, which is also likely.

Comparing interactive relevance feedback techniques in TREC, without controlling results for the influence of relevance assessments by searchers, can lead to faulty conclusions, as illustrated here. Declaring a document to be relevant, when judges disagree, can be particularly damaging, if the false positive assessment occurs in an early iteration of the feedback process and aggressive query expansion is employed. Control can be achieved by statistical strategies, as proposed for the TREC-6 interactive track, or by employing official relevance judgments, as illustrated here.

References

- Bollmann, P., & Wong, S. K. M. (1987). Adaptive linear information retrieval models. *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, 157-163.
- Boyce, B. (1982). Beyond topicality: A two stage view of relevance and the retrieval process. *Information Processing & Management*, 18, 105-109.
- Buckley, C., Salton, G., Allan, J., & Singhal, A. (1995). Automatic query expansion using SMART: TREC 3. In D. K. Harman (Ed.), *Overview of the Third Text REtrieval Conference (TREC-3)* (NIST Spec. Publ. 500-225, pp. 1-19). Washington, DC: U.S. Government Printing Office.

- Buckley, C., Singhal, A., Mitra, M., & Salton, G. (1996). New retrieval approaches using SMART: TREC 4. In D. K. Harman (Ed.), *The Fourth Text REtrieval Conference (TREC-4)* (NIST Spec. Publ. 500-236, pp. 25-48). Washington, DC: U.S. Government Printing Office.
- Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis*. New York: John Wiley & Sons.
- Fishburn, P. C. (1970). *Utility theory for decision making*. New York: John Wiley & Sons.
- Haines, D., & Croft, W. B. (1993). Relevance feedback and inference networks. *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, 2-11.
- Nilsson, N. J. (1965). *Learning machines: Foundations of trainable pattern-classifying systems*. New York: McGraw-Hill.
- Roberts, F. S. (1979). *Measurement theory with applications to decisionmaking, utility, and the social sciences*. Reading, MA: Addison-Wesley.
- Robertson, S. E., & Sparck Jones, K. (1976). Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27, 129-146.
- Robertson, S. E., Walker, S., Beaulieu, M. M., Gatford, M., & Payne, A. (1996). Okapi at TREC-4. In D. K. Harman (Ed.), *The Fourth Text REtrieval Conference (TREC-4)* (NIST Spec. Publ. 500-236, pp. 73-96). Washington, DC: U.S. Government Printing Office.
- Salton, G., & Buckley, C. (1990). Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41, 288-297.
- Salton, G., & McGill, M. J. (1983). *Introduction to modern information retrieval*. New York: McGraw-Hill.
- Shalini, R. (1993). "Citation profiles" to improve relevance in a two-stage retrieval system: A proposal. *Information Processing & Management*, 29, 463-470.
- Shaw, W.M., Jr. (1995). Term-relevance computations and perfect retrieval performance. *Information Processing & Management*, 31, 491-498.
- Shaw, W.M., Jr. (1996). [Letter to the editor]. *Information Processing & Management*, 32, 636-637.
- Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28, 11-21.

- van Rijsbergen, C.J., Harper, D.J., & Porter, M.F. (1981). The selection of good search terms. *Information Processing & Management*, 17, 77-91.
- Wong, S. K. M., & Yao, Y. Y. (1990). Query formulation in linear retrieval models. *Journal of the American Society for Information Science*, 41, 334-341.
- Wong, S. K. M., Yao, Y. Y., & Bollmann, P. (1988). Linear structure in information retrieval. *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, 219-232.
- Wong, S. K. M., Yao, Y. Y., Salton, G., & Buckley, C. (1991). Evaluation of an adaptive linear model. *Journal of the American Society for Information Science*, 42, 723-730.
- Yu, C.T., Buckley, C., Lam, K., & Salton, G. (1983). A generalized term dependence model in information retrieval. *Information Technology: Research and Development*, 2, 129-154.

APPENDIX A

This appendix contains the evaluation results for all TREC-5 runs. The initial pages list each of the runs (identified by the run tags) that were included in the different tasks/tracks. Associated with each tag is the organization that produced the run and additional information such as whether the queries were produced manually or automatically as appropriate. Following the run list is a description of the evaluation measures used for the main tasks and most of the tracks. The confusion, filtering, and interactive tracks use different evaluation measures as described in the respective track reports. The remainder of the appendix contains the evaluation results themselves, in the order given in the run list.

ADHOC RUNS

CATEGORY A DATA

<u>Tag</u>	<u>Organization</u>	<u>Query Construction Method</u>	<u>Topic Length</u>
vtwnA1	Apple Computer	automatic	short
anu5aut1	Australian National University	automatic	short
anu5aut2	Australian National University	automatic	short
city96a2	City University	automatic	short
Cor5A1se	Cornell University	automatic	short
Cor5A2cr	Cornell University	automatic	short
DCU962	Dublin City University	automatic	short
DCU964	Dublin City University	automatic	short
genrl1	GE/Lockheed/NYU/Rutgers	automatic	short
gmu96au1	George Mason University	automatic	short
ibmgd1	IBM Corporation	automatic	short
ibmge1	IBM Corporation	automatic	short
ibms96a	IBM T.J. Watson Research Center	automatic	short
ibms96b	IBM T.J. Watson Research Center	automatic	short
LNaDesc1	Lexis-Nexis	manual	short
LNaDesc2	Lexis-Nexis	manual	short
pircsAAS	Queens College, CUNY	automatic	short
mds002	Royal Melbourne Institute of Technology (RMIT)	automatic	short
ETHas1	Swiss Federal Institute of Technology (ETH)	automatic	short
brkly15	University of California, Berkeley	automatic	short
KUSG2	University of Kansas	automatic	short
KUSG3	University of Kansas	automatic	short
INQ301	University of Massachusetts, Amherst	automatic	short
vtwnB1	Apple Computer	automatic	long
city96a1	City University	automatic	long
genrl2	GE/Lockheed/NYU/Rutgers	automatic	long
gmu96au2	George Mason University	automatic	long
pircsAAL	Queens College, CUNY	automatic	long
mds001	Royal Melbourne Institute of Technology (RMIT)	automatic	long
ETHal1	Swiss Federal Institute of Technology (ETH)	automatic	long
brkly16	University of California, Berkeley	automatic	long
INQ302	University of Massachusetts, Amherst	automatic	long
anu5man4	Australian National University	manual	
anu5man6	Australian National University	manual	
CLCLUS	CLARITECH Corporation	manual	
CLTHES	CLARITECH Corporation	manual	
Cor5M1le	Cornell University	manual	
Cor5M2rf	Cornell University	manual	
DCU961	Dublin City University	manual	
DCU963	Dublin City University	manual	
fsclt3	FS Consulting	manual	
fsclt4	FS Consulting	manual	
genrl3	GE/Lockheed/NYU/Rutgers	manual	
genrl4	GE/Lockheed/NYU/Rutgers	manual	

ADHOC RUNS (Continued)

CATEGORY A DATA (Continued)

<u>Tag</u>	<u>Organization</u>	<u>Query Method</u>
gmu96ma1	George Mason University	manual
gmu96ma2	George Mason University	manual
erliA1	GSI-Erli	manual
ibmgd2	IBM Corporation	manual
ibmge2	IBM Corporation	manual
LNmFull1	Lexis-Nexis	manual
LNmFull2	Lexis-Nexis	manual
colm1	Open Text Corporation	manual
colm4	Open Text Corporation	manual
pircsAM1	Queens College, CUNY	manual
pircsAM2	Queens College, CUNY	manual
mds003	Royal Melbourne Institute of Technology (RMIT)	manual
ETHme1	Swiss Federal Institute of Technology (ETH)	manual
brkly17	University of California, Berkeley	manual
brkly18	University of California, Berkeley	manual
uwgcx0	University of Waterloo	manual
uwgcx1	University of Waterloo	manual

CATEGORY B DATA

<u>Tag</u>	<u>Organization</u>	<u>Query Construction Method</u>	<u>Topic Length</u>
Ctifr1	Computer Technology Institute	automatic	short
Ctifr2	Computer Technology Institute	automatic	short
DCU969	Dublin City University	automatic	short
DCU96B	Dublin City University	automatic	short
DCU96D	Dublin City University	automatic	short
Mercure-as	Institut de Recherche en Informatique de Toulouse	automatic	short
MONASH	Monash University	automatic	short
UniNE7	Universite de Neuchatel	automatic	short
sdmix2	University of California, San Diego	automatic	short
glair4	University of Glasgow	automatic	short
umcpa1	University of Maryland	automatic	short
Mercure-al	Institut de Recherche en Informatique de Toulouse	automatic	long
UniNE8	Universite de Neuchatel	automatic	long
sdmix1	University of California, San Diego	automatic	long
DCU968	Dublin City University	manual	
DCU96A	Dublin City University	manual	
DCU96C	Dublin City University	manual	
uncis1	University of North Carolina	manual	
uncis2	University of North Carolina	manual	

ROUTING RUNS

CATEGORY A DATA

<u>Tag</u>	<u>Organization</u>	<u>Query Construction Method</u>
city96r1	City University	automatic
city96r2	City University	automatic
Cor5R1cc	Cornell University	automatic
Cor5R2cr	Cornell University	automatic
genrl5	GE/Lockheed/NYU/Rutgers	automatic
genrl6	GE/Lockheed/NYU/Rutgers	automatic
itidp1	Information Technology Institute	automatic
itidp2	Information Technology Institute	automatic
nmsu_1	New Mexico State University	automatic
pircsg0	Queens College, CUNY	automatic
pircsg6	Queens College, CUNY	automatic
rutAPglob	Rutgers University	automatic
rutAPspt	Rutgers University	automatic
ETHru1	Swiss Federal Institute of Technology (ETH)	automatic
brkly13	University of California, Berkeley	automatic
brkly14	University of California, Berkeley	automatic
ispaR	University of Illinois	automatic
INQ303	University of Massachusetts, Amherst	automatic
xerox.rout1	Rank Xerox Research Center	automatic
xerox.rout2	Rank Xerox Research Center	automatic
xerox.rout3	Rank Xerox Research Center	automatic
erliR1	GSI-Erli	manual
uwgcr0	University of Waterloo	manual

CATEGORY B DATA

<u>Tag</u>	<u>Organization</u>	<u>Query Construction Method</u>
Mercure-r1	Institut de Recherche en Informatique de Toulouse	automatic
Mercure-r2	Institut de Recherche en Informatique de Toulouse	automatic
sdmix3	University of California, San Diego	automatic

TRACKS

SPANISH

<u>Tag</u>	<u>Organization</u>	<u>Query Construction Method</u>	<u>Topic Length</u>
Cor5SP1s	Cornell University	automatic	short
DCU966	Dublin City University	automatic	short
gmu96sp1	George Mason University	automatic	short
gmu96sp2	George Mason University	automatic	short
nmsuc1	New Mexico State University	automatic	short
nmsuc2	New Mexico State University	automatic	short
nmsuc3	New Mexico State University	automatic	short
BrklySP5	University of California, Berkeley	automatic	short
SIN300	University of Massachusetts	automatic	short
xerox-spD	Rank Xerox Research Center	automatic	short
Cor5SP2l	Cornell University	automatic	long
SIN301	University of Massachusetts	automatic	long
xerox-spP	Rank Xerox Research Center	automatic	long
xerox-spS	Rank Xerox Research Center	automatic	long
xerox-spT	Rank Xerox Research Center	automatic	long
DCU965	Dublin City University	manual	
DCU967	Dublin City University	manual	
BrklySP6	University of California, Berkeley	manual	

CHINESE

<u>Tag</u>	<u>Organization</u>	<u>Query Construction Method</u>
city96c1	City University	automatic
city96c2	City University	automatic
CLCHNA	CLARITECH Corporation	automatic
Cor5C1vt	Cornell University	automatic
Cor5C2ex	Cornell University	automatic
gmu96ca1	George Mason University	automatic
gmu96ca2	George Mason University	automatic
itcn1	Information Technology Institute	automatic
pircsCw	Queens College, CUNY	automatic
pircsCwc	Queens College, CUNY	automatic
mds004	Royal Melbourne Institute of Technology	automatic
mds005	Royal Melbourne Institute of Technology	automatic
BrklyCH1	University of California, Berkeley	automatic
HIN300	University of Massachusetts	automatic
HIN301	University of Massachusetts	automatic
CLCHNM	CLARITECH Corporation	manual
gmu96cm1	George Mason University	manual
gmu96cm2	George Mason University	manual
itcn2	Information Technology Institute	manual
BrklyCH2	University of California, Berkeley	manual

FILTERING

<u>Tag</u>	<u>Organization</u>
city96f	City University
INR3	University of Massachusetts, Amherst
INTXA	InText Systems
INTXM	InText Systems
ispF	University of Illinois
iti96f	Information Technology Institute
pircs96f	Queens University, CUNY
xerox.f1	Rank Xerox Research Center
xerox.f2	Rank Xerox Research Center
xerox.f3	Rank Xerox Research Center

DATABASE MERGING CATEGORY A

<u>Tag</u>	<u>Organization</u>	<u>Query Construction Method</u>
anu5mrg0	Australian National University	manual
anu5mrg1	Australian National University	manual
anu5mrg7	Australian National University	manual
fsclt3m	FS Consulting	manual

DATABASE MERGING CATEGORY B

<u>Tag</u>	<u>Organization</u>	<u>Query Construction Method</u>
UniNE0	Universite de Neuchatel	automatic
UniNE9	Universite de Neuchatel	automatic

INTERACTIVE

Organization

City University
Rutgers University

NLP

<u>Tag</u>	<u>Organization</u>
CLATMC	CLARITECH Corporation
CLATMN	CLARITECH Corporation
CLPHR0	CLARITECH Corporation
CLPHR1	CLARITECH Corporation
CLPHR2	CLARITECH Corporation
genlp1	GE/Lockheed/NYU/Rutgers
genlp2	GE/Lockheed/NYU/Rutgers
genlp3	GE/Lockheed/NYU/Rutgers
genlp4	GE/Lockheed/NYU/Rutgers
sbase1.new	GE/Lockheed/NYU/Rutgers
sbase2.new	GE/Lockheed/NYU/Rutgers
MTRa961	MITRE
xerox_nlp1	Rank Xerox Research Center
xerox_nlp2	Rank Xerox Research Center
xerox_nlp3	Rank Xerox Research Center
xerox_nlp4	Rank Xerox Research Center
xerox_nlp5	Rank Xerox Research Center
xerox_nlp6	Rank Xerox Research Center

CONFUSION

<u>Tag</u>	<u>Organization</u>
anu5con	Australian National University
CLCON	CLARITECH Corporation
CLCONF	CLARITECH Corporation
gmu961	George Mason University
gmu962	George Mason University
rutcf	Rutgers University
ETHFR94N	Swiss Federal Institute of Technology (ETH)
ETHFR94P	Swiss Federal Institute of Technology (ETH)

Evaluation Techniques and Measures

Categories

The results following this section are organized according to the task accomplished by the run: ad hoc, routing, or a track task.

I. Ad hoc

Retrieval using an “ad hoc” topic such as a researcher might use in a library environment. In TREC this implies that the input topic has no training material such as relevance judgments to aid in the construction of the input query.

A. Category A

Systems running TREC topics against all documents from TREC Disks 2 and 4.

B. Category B

Systems running TREC topics against the Wall Street Journal (WSJ) on TREC Disk 2. (Intended for new groups, allowing them to scale their systems to handle large collections.)

II. Routing

Retrieval using a “routing” query such as a profile to filter some incoming document stream. In TREC this implies that the input topic has training material, including relevance judgments against the training documents, to use in constructing the input query or profile. This query is then used against new documents (the test documents).

A. Category A

Systems running TREC topics against a set of Foreign Broadcast Information Service (FBIS) documents.

B. Category B

Systems running TREC topics against FBIS documents contained in files beginning with ‘FB3’. (Intended for new groups, allowing them to scale their systems to handle large collections.)

Evaluation Measures

I. Recall

A measure of the ability of a system to present all relevant items.

$$\text{recall} = \frac{\text{number of relevant items retrieved}}{\text{number of relevant items in collection}}$$

II. Precision.

A measure of the ability of a system to present only relevant items.

$$\text{precision} = \frac{\text{number of relevant items retrieved}}{\text{total number of items retrieved}}$$

System Results Description

Each of the following pages contains the evaluation results for one run. A page is comprised of a header (containing the task and organization name), 3 tables, and 2 graphs.

Tables

Tables are generated by *trec_eval* courtesy of Chris Buckley using the SMART methodology as defined in Salton and McGill [1].

I. "Summary Statistics" Table

Table 1 is a sample "Summary Statistics" Table

Table 1: Sample "Summary Statistics" Table.

Summary Statistics	
Run	Cor5A2cr-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2848

A. Run

A description of the run. It contains the run tag provided by the participant, and as applicable, whether the run is Category A or B, whether queries were constructed manually or automatically, and whether long or short topic descriptions were used.

B. Number of Topics

Number of topics searched in this run (generally 50 topics are run for each task).

C. Total number of documents over all topics (the number of topics given in B).

i. Retrieved

Number of documents submitted to NIST. This is usually 50,000 (50 topics \times 1000 documents), but is less when fewer than 1000 documents are retrieved per topic.

ii. Relevant

Total possible relevant documents within a given task and category.

iii. Rel_{ret}

Total number of relevant documents returned by a run over all the topics.

II. "Recall Level Precision Averages" Table.

Table 2 is a sample "Recall Level Precision Averages" Table.

Table 2: Sample "Recall Level Precision Averages" Table.

Recall Level Precision Averages	
Recall	Precision
0.00	0.5857
0.10	0.3927
0.20	0.3252
0.30	0.2799
0.40	0.2521
0.50	0.2131
0.60	0.1776
0.70	0.1395
0.80	0.0885
0.90	0.0415
1.00	0.0118
Average precision over all relevant docs	
non-interpolated	0.2109

A. Precision at 11 standard recall levels

The precision averages at 11 standard recall levels are used to compare the performance of different systems and as the input for plotting the recall-precision graph (see below). Each recall-precision average is computed by summing the precisions at the specified recall cutoff value (denoted by $\sum P_\lambda$ where P_λ is the precision at recall level λ) and then dividing by the number of topics.

$$\frac{\sum_{i=1}^{NUM} P_\lambda}{NUM} \quad \lambda = \{0.0, 0.1, 0.2, 0.3, \dots, 1.0\}$$

- Interpolating recall-precision

Standard recall levels facilitate averaging and plotting retrieval results. Since the actual recall levels of individual queries are seldom equal to the standard levels, interpolation is used to define the precision at those points. The interpolated precision at the i th recall level (R^i) is defined to be the maximum precision at all points p such that $R^{i-1} \leq p \leq R^i$.

For example, if there are only 3 relevant documents, and these are retrieved at ranks 4, 9, and 20, then the exact recall points are 0.33, 0.67, and 1.0. Interpolated precisions are computed using the true recall values (precision 0.25 at recall 0.33, precision 0.22 at recall 0.67, and precision 0.15 at recall 1.0, respectively) and mapping them to the 11 recall cutoff values using the above rule. Therefore, the precisions at recall points 0.0, 0.1, 0.2, 0.3 are 0.25, the precision at recall points 0.4, 0.5 0.6, are 0.22 and precision at recall points 0.7, 0.8, 0.9, 1.0 are 0.15. Note that theoretically precision is not defined at a recall of 0.0, however this interpolation rule allows values to be determined.

B. Average precision over all relevant documents, non-interpolated

This is a single-valued measure that reflects the performance over all relevant documents. It rewards systems that retrieve relevant documents quickly (highly ranked).

The measure is not an average of the precision at standard recall levels. Rather, it is the average of the precision value obtained after each relevant document is retrieved. As an example, consider a query that has four relevant documents which are retrieved at ranks 1, 2, 4, and 7. The actual precision obtained when each relevant document is retrieved is 1, 1, 0.75, and 0.57, respectively, the mean of which is 0.83. Thus, the average precision over all relevant documents for this query is 0.83.

III. "Document Level Averages" Table

Table 3 is a sample "Document Level Averages" Table.

Table 3: Sample "Document Level Averages" Table.

Document Level Averages	
	Precision
At 5 docs	0.4240
At 10 docs	0.3800
At 15 docs	0.3453
At 20 docs	0.3270
At 30 docs	0.2913
At 100 docs	0.2018
At 200 docs	0.1544
At 500 docs	0.0933
At 1000 docs	0.0570
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2404

A. Precision at 9 document cutoff values

The precision computed after a given number of documents have been retrieved reflects the actual measured system performance as a user might see it. Each document precision average is computed by summing the precisions at the specified document cutoff value and dividing by the number of topics (50).

B. R-Precision

R-Precision is the precision after R documents have been retrieved, where R is the number of relevant documents for the topic. It de-emphasizes the exact ranking of the retrieved relevant documents, which can be particularly useful in TREC where there are large numbers of relevant documents.

The average R-Precision for a run is computed by taking the mean of the R-Precisions of the individual topics in the run. For example, assume a run consists of two topics, one with 50 relevant documents and another with 10 relevant documents. If the retrieval system returns 17 relevant documents in the top 50 documents for the first topic, and 7 relevant documents in the top 10 for the second topic, then the run's R-Precision would be $\frac{\frac{17}{50} + \frac{7}{10}}{2}$ or 0.52.

Graphs

I. Recall-Precision Graph

Figure 1 is a sample Recall-Precision Graph.

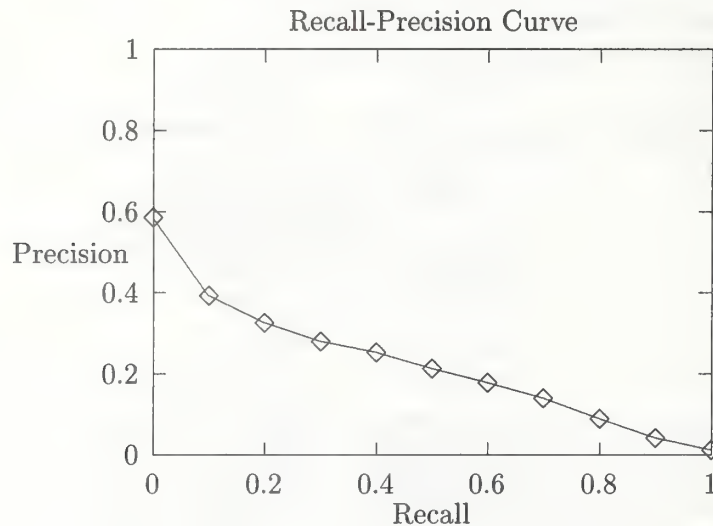


Figure 1: Sample Recall-Precision Graph.

The Recall-Precision Graph is created using the 11 cutoff values from the Recall Level Precision Averages. Typically these graphs slope downward from left to right, enforcing the notion that as more relevant documents are retrieved (recall increases), the more nonrelevant documents are retrieved (precision decreases).

This graph is the most commonly used method for comparing systems. The plots of different runs can be superimposed on the same graph to determine which run is superior. Curves closest to the upper right-hand corner of the graph (where recall and precision are maximized) indicate the best performance. Comparisons are best made in three different recall ranges: 0

to 0.2, 0.2 to 0.8, and 0.8 to 1. These ranges characterize high precision, middle recall, and high recall performance, respectively.

II. Average Precision Histogram.

Figure 2 is a sample Average Precision Histogram.

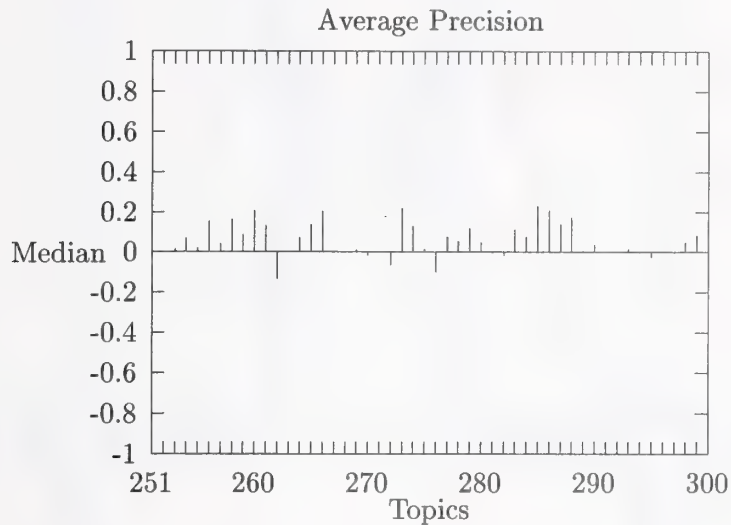


Figure 2: Sample Average Precision Histogram.

The Average Precision Histogram measures the precision of a run on each topic against the median precision of all corresponding runs on that topic. This graph is intended to give insight into the performance of individual systems and the types of topics that they handle well.

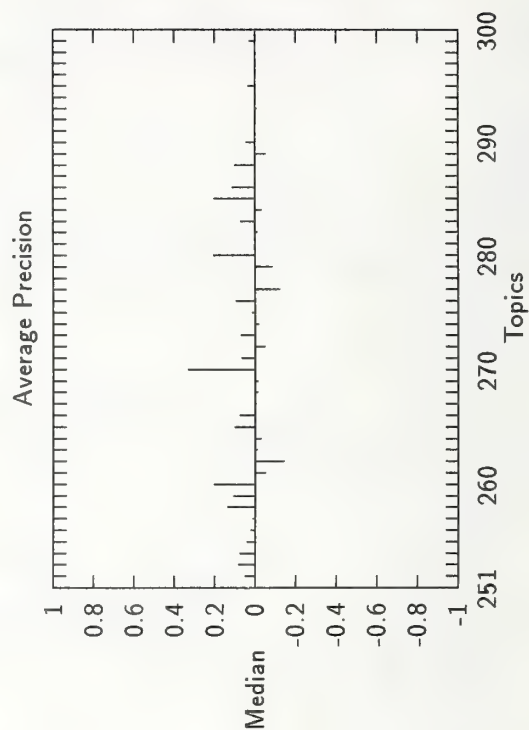
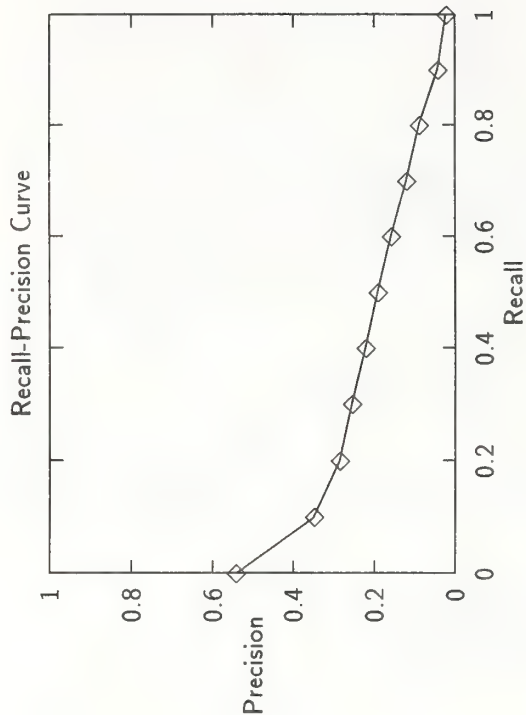
References

- [1] G. SALTON AND M. MCGILL, *Introduction to Modern Information Retrieval*, McGraw-Hill, New York, first ed., 1983.

Summary Statistics	
Run Number	vtwnA1—category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2511

Recall Level Precision Averages	
Recall	Precision
0.00	0.5417
0.10	0.3497
0.20	0.2854
0.30	0.2559
0.40	0.2228
0.50	0.1921
0.60	0.1590
0.70	0.1213
0.80	0.0889
0.90	0.0436
1.00	0.0225
Average precision over all relevant docs	
non-interpolated	0.1894

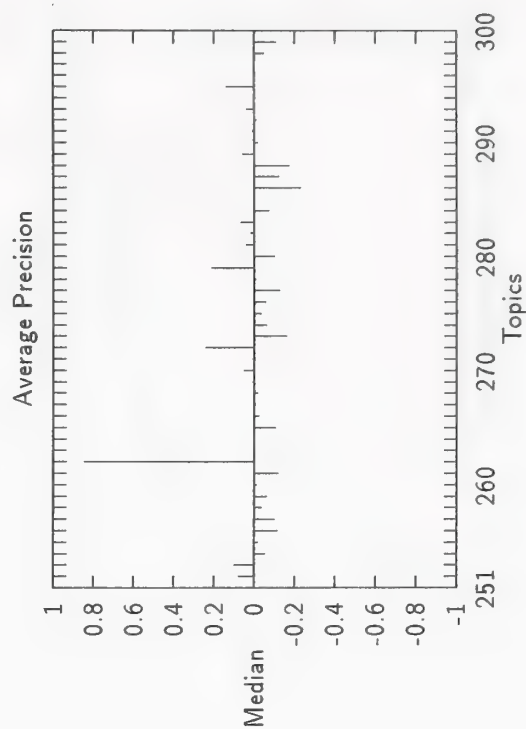
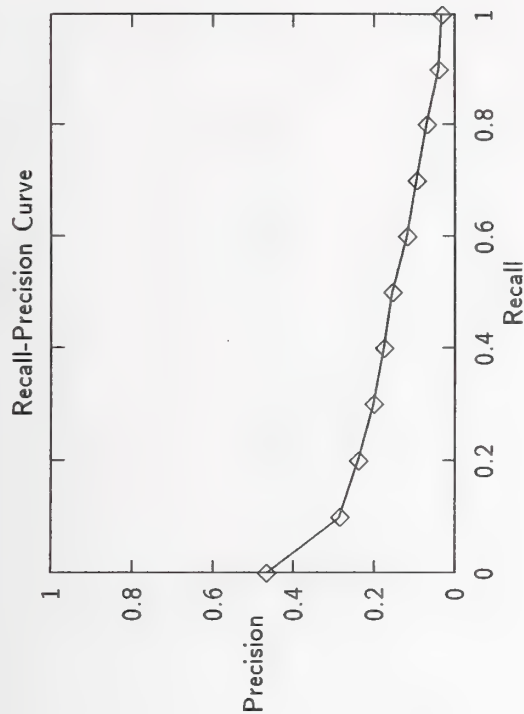
Document Level Averages	
	Precision
At 5 docs	0.3360
At 10 docs	0.3140
At 15 docs	0.2933
At 20 docs	0.2670
At 30 docs	0.2527
At 100 docs	0.1728
At 200 docs	0.1331
At 500 docs	0.0807
At 1000 docs	0.0502
R—Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2315



Summary Statistics		
Run Number	anu5aut1-category A, automatic, short topic	
Number of Topics	50	
Total number of documents over all topics		
Retrieved:	50000	
Relevant:	5524	
Rel_ret:	1879	

Recall Level Precision Averages	
Recall	Precision
0.00	0.4679
0.10	0.2869
0.20	0.2399
0.30	0.2027
0.40	0.1760
0.50	0.1557
0.60	0.1191
0.70	0.0954
0.80	0.0703
0.90	0.0390
1.00	0.0316
Average precision over all relevant docs	
non-interpolated	0.1537

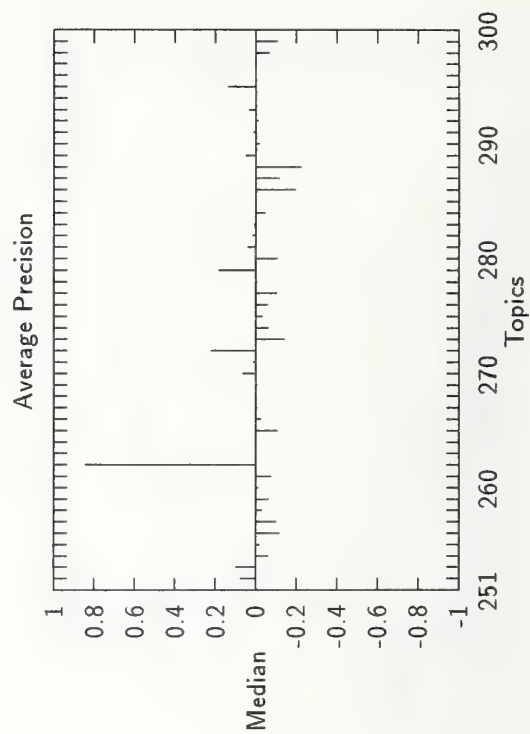
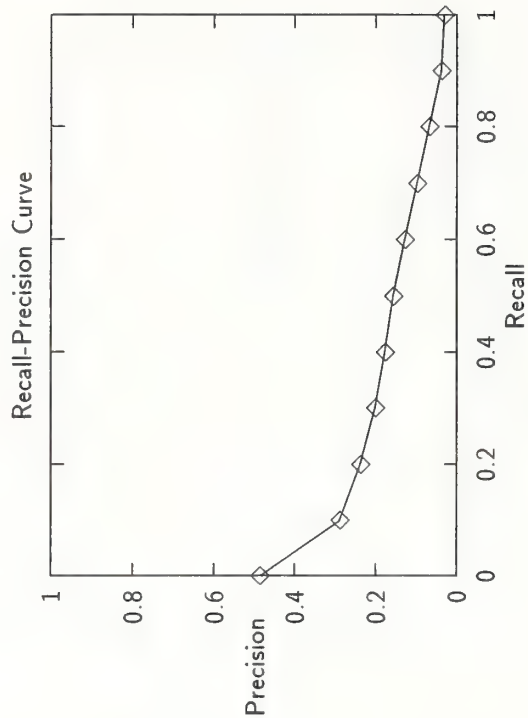
Document Level Averages	
	Precision
At 5 docs	0.2520
At 10 docs	0.2540
At 15 docs	0.2240
At 20 docs	0.2160
At 30 docs	0.1893
At 100 docs	0.1384
At 200 docs	0.0929
At 500 docs	0.0593
At 1000 docs	0.0376
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1948



Summary Statistics	
Run Number	anu5aut2-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	1987

Recall Level Precision Averages	
Recall	Precision
0.00	0.4858
0.10	0.2898
0.20	0.2378
0.30	0.2014
0.40	0.1778
0.50	0.1575
0.60	0.1286
0.70	0.0987
0.80	0.0687
0.90	0.0381
1.00	0.0303
Average precision over all relevant docs	
non-interpolated	0.1538

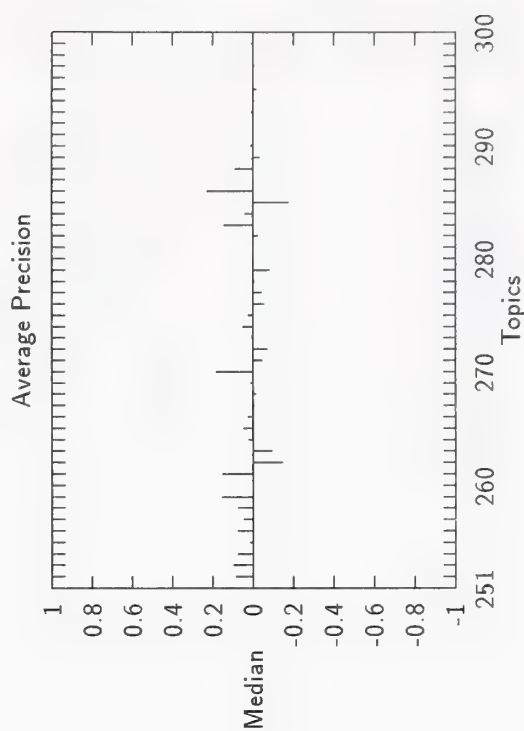
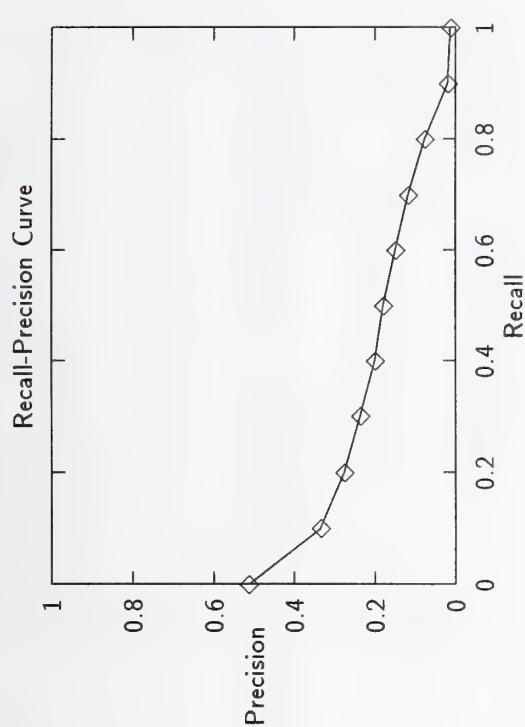
Document Level Averages	
At 5 docs	0.2400
At 10 docs	0.2280
At 15 docs	0.2107
At 20 docs	0.2100
At 30 docs	0.1887
At 100 docs	0.1378
At 200 docs	0.0979
At 500 docs	0.0622
At 1000 docs	0.0397
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1980



Summary Statistics	
Run Number	city96a2-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2592

Recall Level Precision Averages	
Recall	Precision
0.00	0.5139
0.10	0.3361
0.20	0.2766
0.30	0.2365
0.40	0.2004
0.50	0.1795
0.60	0.1489
0.70	0.1177
0.80	0.0757
0.90	0.0193
1.00	0.0135
Average precision over all relevant docs	
non-interpolated	0.1751

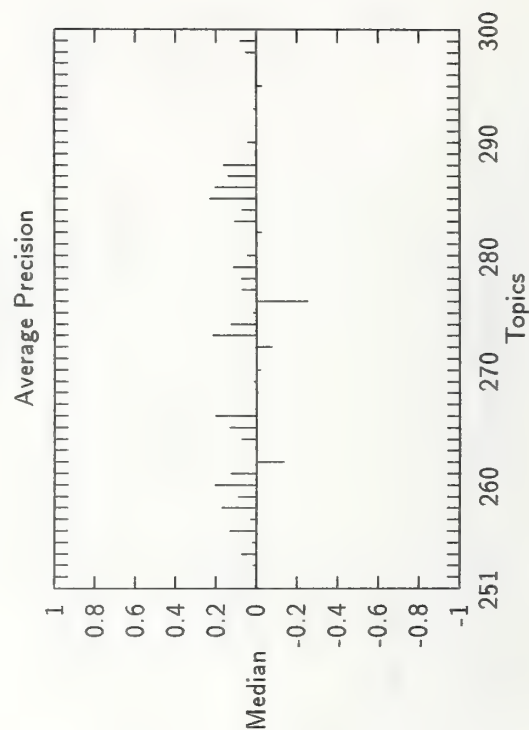
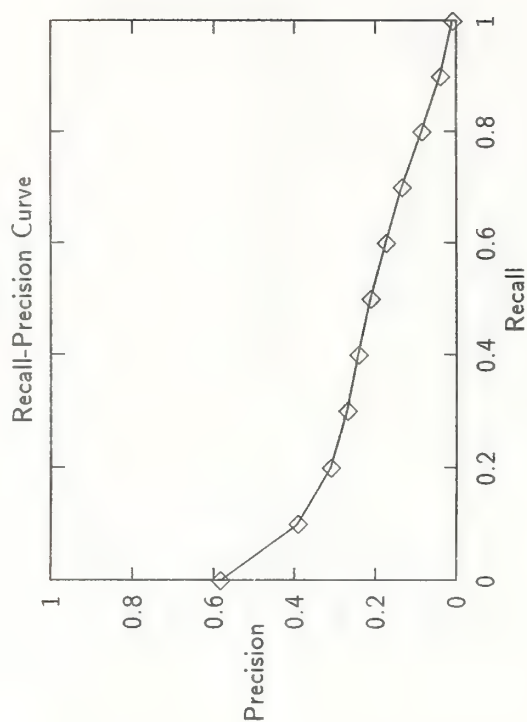
Document Level Averages	
At 5 docs	0.3360
At 10 docs	0.2860
At 15 docs	0.2973
At 20 docs	0.2840
At 30 docs	0.2600
At 100 docs	0.1784
At 200 docs	0.1355
At 500 docs	0.0822
At 1000 docs	0.0518
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2134



Summary Statistics	
Run Number	Cor5A1se-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2849

Recall Level Precision Averages	
Recall	Precision
0.00	0.5841
0.10	0.3933
0.20	0.3110
0.30	0.2703
0.40	0.2426
0.50	0.2126
0.60	0.1742
0.70	0.1358
0.80	0.0853
0.90	0.0401
1.00	0.0102
Average precision over all relevant docs	
non-interpolated	0.2065

Document Level Averages	
	Precision
At 5 docs	0.4160
At 10 docs	0.3760
At 15 docs	0.3533
At 20 docs	0.3260
At 30 docs	0.2880
At 100 docs	0.2002
At 200 docs	0.1540
At 500 docs	0.0932
At 1000 docs	0.0570
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2331

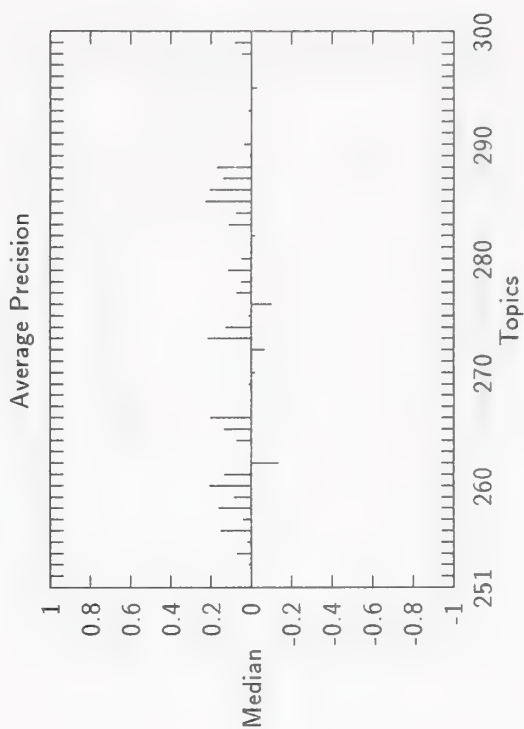
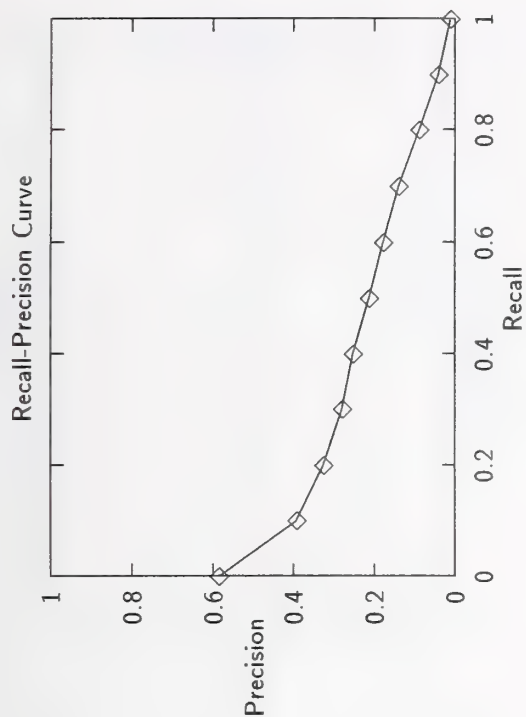


Summary Statistics		
Run Number	Cor5A2cr-category A, automatic, short topic	
Number of Topics	50	
Total number of documents over all topics		
Retrieved:	50000	
Relevant:	5524	
Rel_ret:	2848	

Recall Level Precision Averages	
Recall	Precision
0.00	0.5857
0.10	0.3927
0.20	0.3252
0.30	0.2799
0.40	0.2521
0.50	0.2131
0.60	0.1776
0.70	0.1395
0.80	0.0885
0.90	0.0415
1.00	0.0118

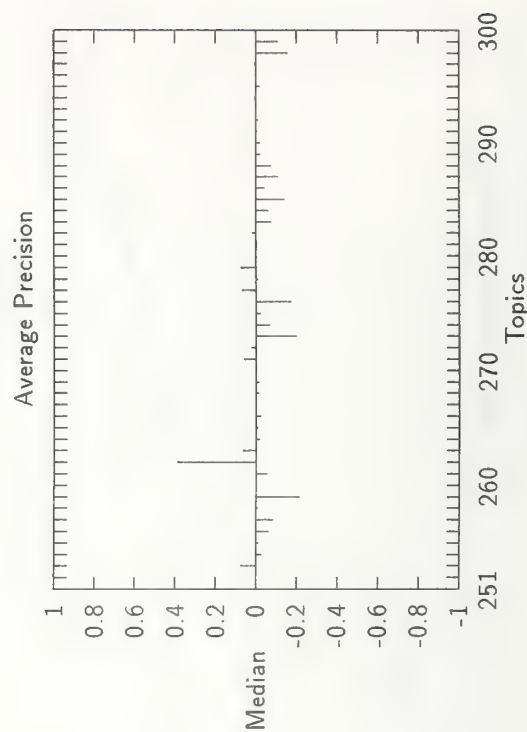
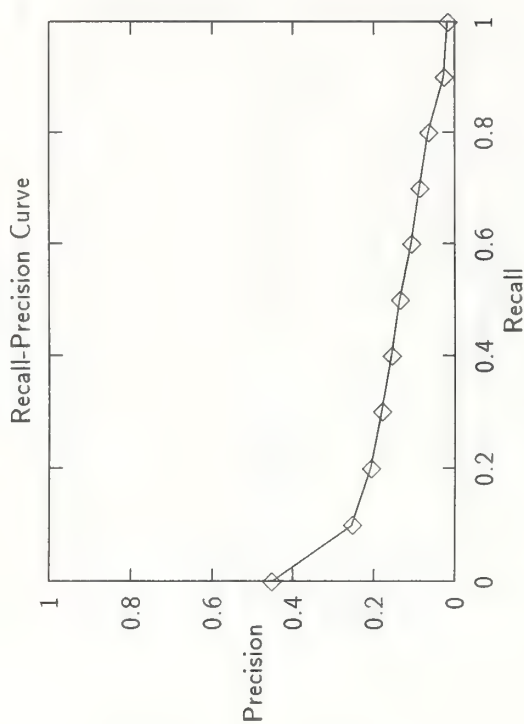
Average precision over all relevant docs	
non-interpolated	0.2109

Document Level Averages	
At 5 docs	0.4240
At 10 docs	0.3800
At 15 docs	0.3453
At 20 docs	0.3270
At 30 docs	0.2913
At 100 docs	0.2018
At 200 docs	0.1544
At 500 docs	0.0933
At 1000 docs	0.0570
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2404



Summary Statistics	
Run Number	DCU962-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	1943

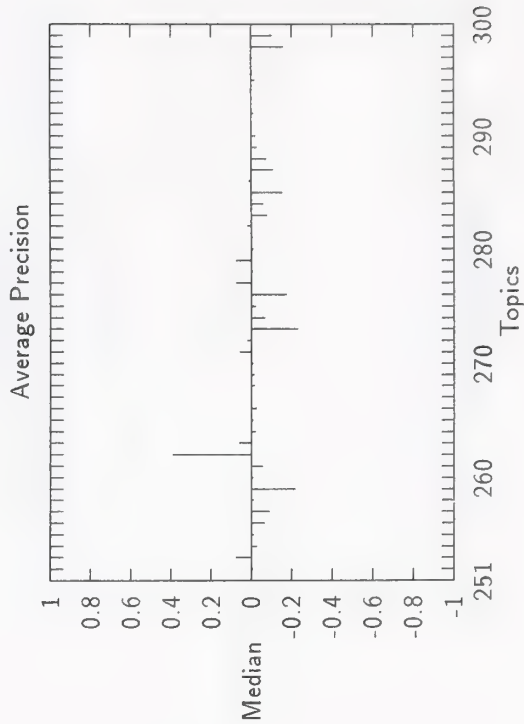
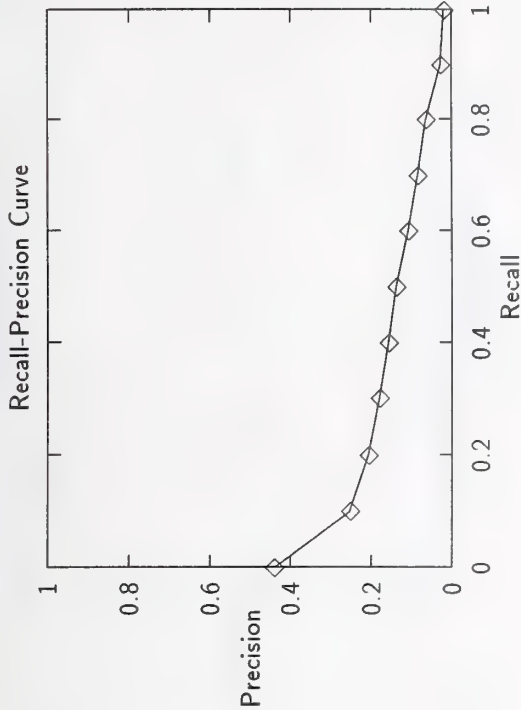
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.4525	At 5 docs	0.2800
0.10	0.2540	At 10 docs	0.2540
0.20	0.2076	At 15 docs	0.2280
0.30	0.1805	At 20 docs	0.2090
0.40	0.1575	At 30 docs	0.1867
0.50	0.1374	At 100 docs	0.1236
0.60	0.1079	At 200 docs	0.0896
0.70	0.0878	At 500 docs	0.0578
0.80	0.0644	At 1000 docs	0.0389
0.90	0.0277	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0181		
Average precision over all relevant docs			
non-interpolated	0.1340	Exact	0.1721



Summary Statistics	
Run Number	DCU964-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	1940

Recall Level Precision Averages	
Recall	Precision
0.00	0.4404
0.10	0.2531
0.20	0.2067
0.30	0.1802
0.40	0.1571
0.50	0.1380
0.60	0.1085
0.70	0.0843
0.80	0.0639
0.90	0.0275
1.00	0.0192
Average precision over all relevant docs	
non-interpolated	0.1334

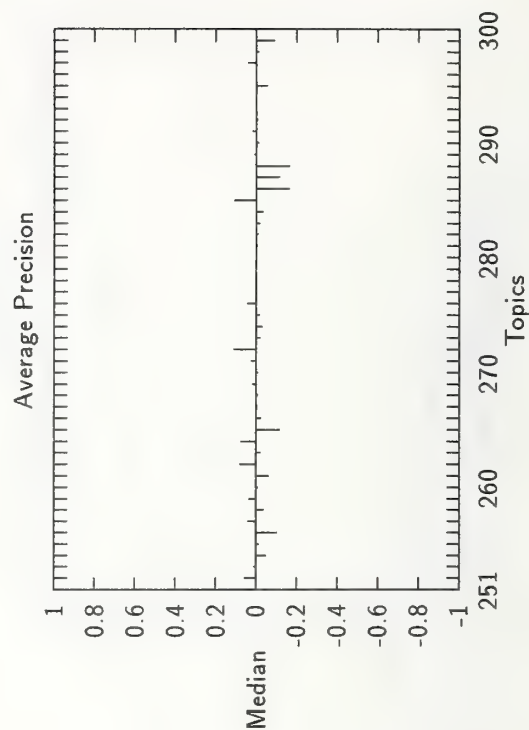
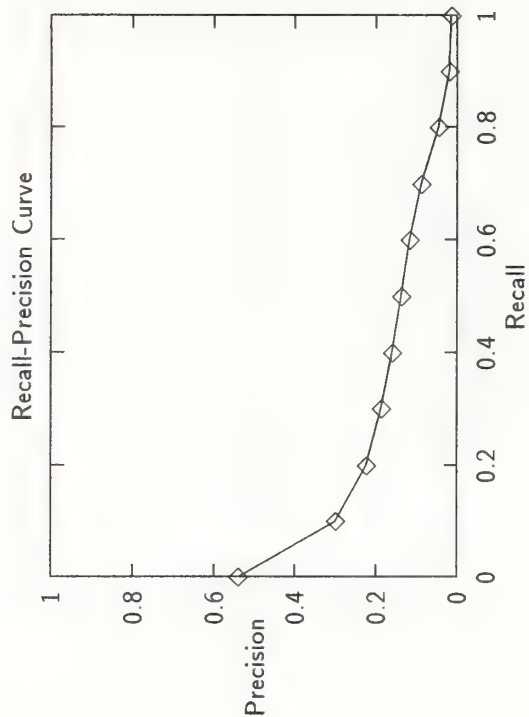
Document Level Averages	
	Precision
At 5 docs	0.2840
At 10 docs	0.2460
At 15 docs	0.2293
At 20 docs	0.2050
At 30 docs	0.1880
At 100 docs	0.1242
At 200 docs	0.0889
At 500 docs	0.0568
At 1000 docs	0.0388
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1695



Summary Statistics		
Run Number	genrl1-category A, automatic, short topic	50
Number of Topics		
Total number of documents over all topics		
Retrieved:	50000	
Relevant:	5524	
Rel_ret:	2313	

Recall Level Precision Averages		
Recall	Precision	
0.00	0.5405	
0.10	0.3010	
0.20	0.2248	
0.30	0.1877	
0.40	0.1610	
0.50	0.1379	
0.60	0.1181	
0.70	0.0896	
0.80	0.0462	
0.90	0.0184	
1.00	0.0140	
Average precision over all relevant docs		
non-interpolated	0.1460	

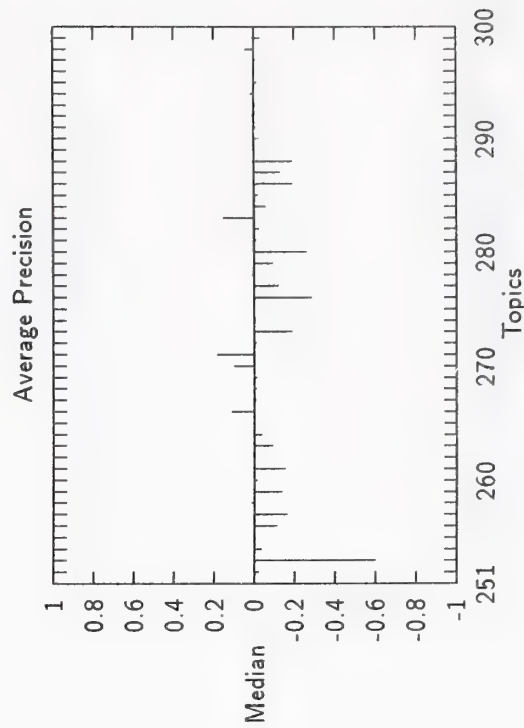
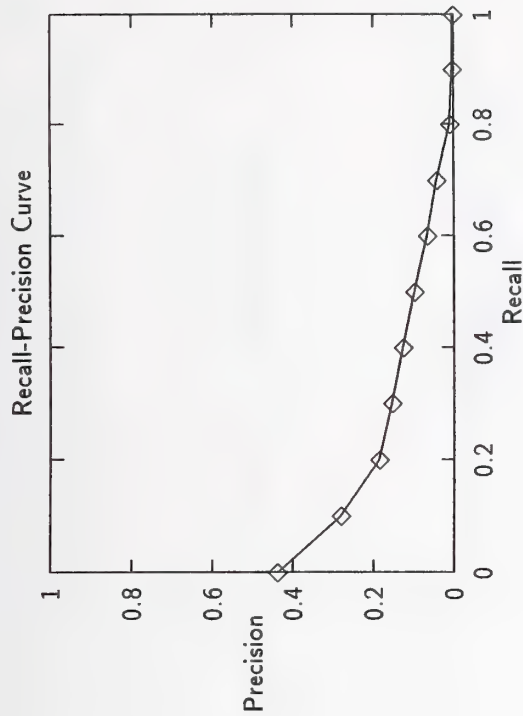
Document Level Averages	
	Precision
At 5 docs	0.3480
At 10 docs	0.2900
At 15 docs	0.2667
At 20 docs	0.2620
At 30 docs	0.2420
At 100 docs	0.1594
At 200 docs	0.1141
At 500 docs	0.0693
At 1000 docs	0.0463
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1847



Summary Statistics	
Run Number	gmu96au1-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	1876

Recall Level Precision Averages	
Recall	Precision
0.00	0.4392
0.10	0.2808
0.20	0.1854
0.30	0.1538
0.40	0.1271
0.50	0.0992
0.60	0.0664
0.70	0.0434
0.80	0.0090
0.90	0.0031
1.00	0.0020
Average precision over all relevant docs	
non-interpolated	0.1079

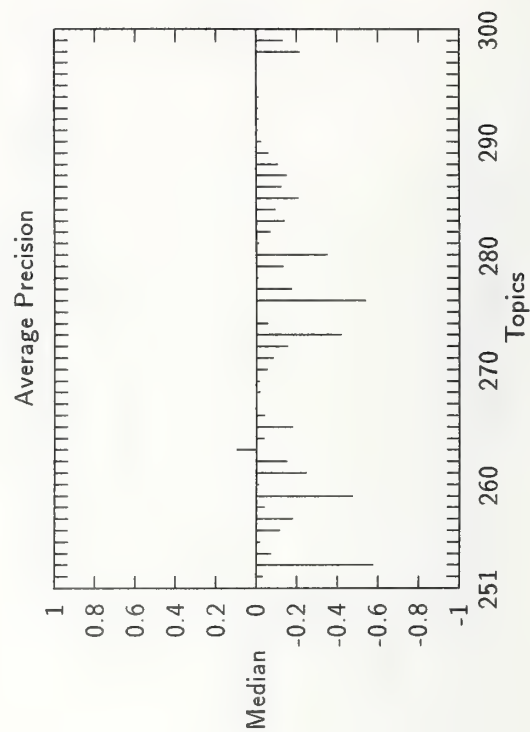
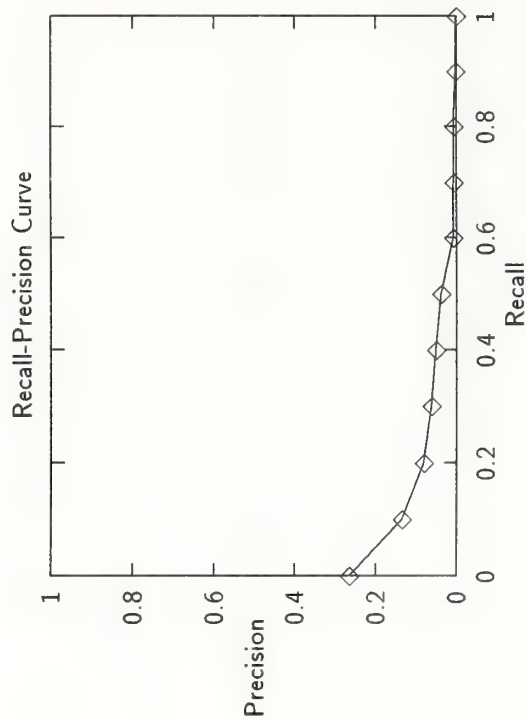
Document Level Averages	
At 5 docs	0.2680
At 10 docs	0.2300
At 15 docs	0.2240
At 20 docs	0.2050
At 30 docs	0.1833
At 100 docs	0.1232
At 200 docs	0.0946
At 500 docs	0.0579
At 1000 docs	0.0375
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1451



Summary Statistics	
Run Number	ibmgdl-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	657

Recall Level Precision Averages	
Recall	Precision
0.00	0.2660
0.10	0.1354
0.20	0.0806
0.30	0.0619
0.40	0.0515
0.50	0.0381
0.60	0.0079
0.70	0.0066
0.80	0.0062
0.90	0.0019
1.00	0.0001
Average precision over all relevant docs	
non-interpolated	0.0462

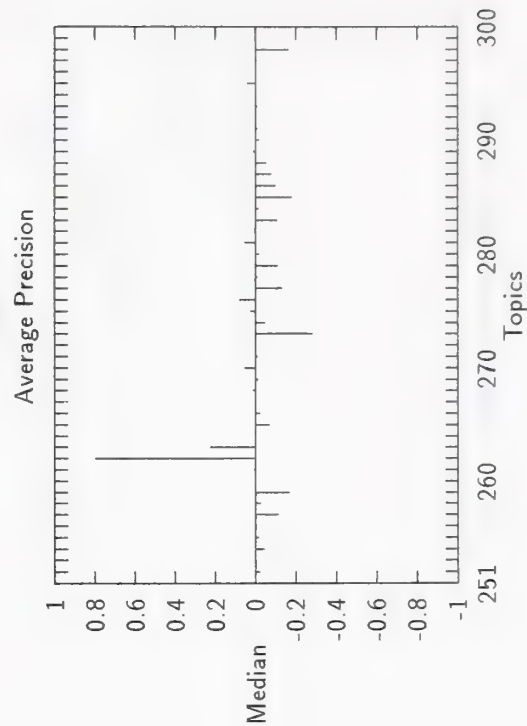
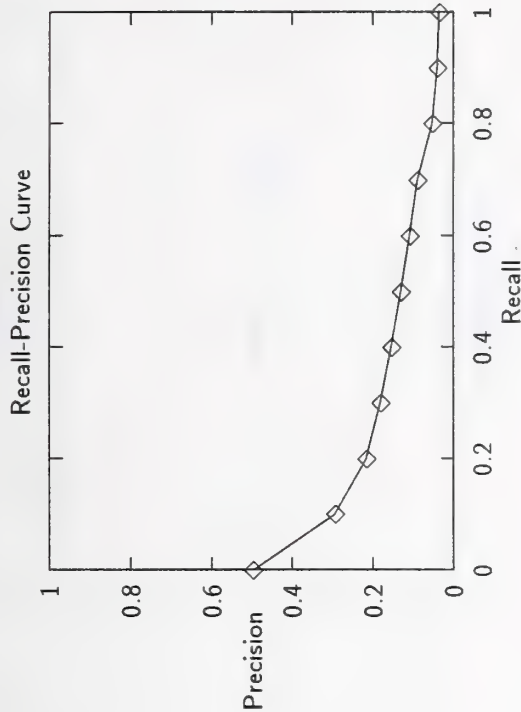
Document Level Averages	
At 5 docs	0.1560
At 10 docs	0.1100
At 15 docs	0.0960
At 20 docs	0.0970
At 30 docs	0.0827
At 100 docs	0.0554
At 200 docs	0.0365
At 500 docs	0.0200
At 1000 docs	0.0131
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0686



Summary Statistics		
Run Number	ibmge1-category A, automatic, short topic	
Number of Topics	50	
Total number of documents over all topics		
Retrieved:	50000	
Relevant:	5524	
Rel_ret:	1764	

Recall Level Precision Averages	
Recall	Precision
0.00	0.4974
0.10	0.2947
0.20	0.2157
0.30	0.1818
0.40	0.1548
0.50	0.1307
0.60	0.1089
0.70	0.0883
0.80	0.0526
0.90	0.0396
1.00	0.0348
Average precision over all relevant docs	
non-interpolated	0.1451

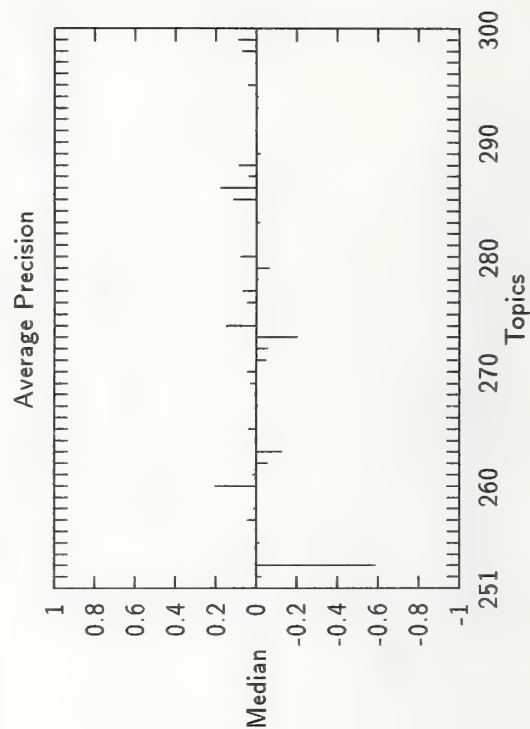
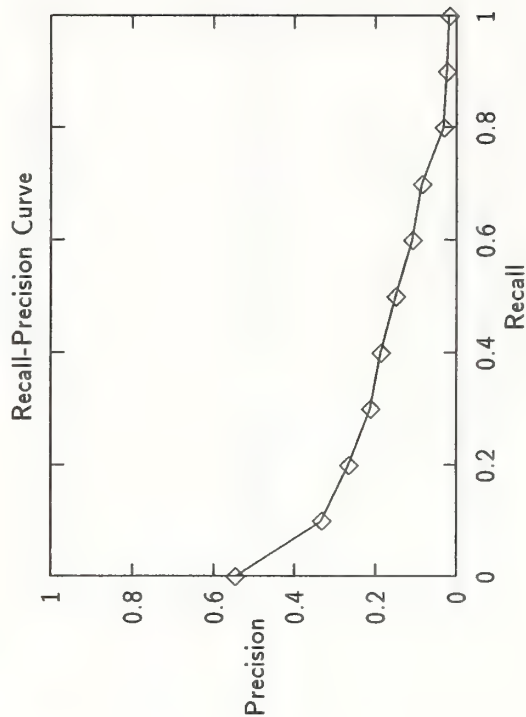
Document Level Averages	
	Precision
At 5 docs	0.3320
At 10 docs	0.2740
At 15 docs	0.2413
At 20 docs	0.2170
At 30 docs	0.1987
At 100 docs	0.1310
At 200 docs	0.0887
At 500 docs	0.0532
At 1000 docs	0.0353
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1811



Summary Statistics	
Run Number	ibms96a--category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel.ret:	2114

Recall Level Precision Averages	
Recall	Precision
0.00	0.5472
0.10	0.3343
0.20	0.2666
0.30	0.2137
0.40	0.1880
0.50	0.1511
0.60	0.1091
0.70	0.0853
0.80	0.0321
0.90	0.0244
1.00	0.0175
Average precision over all relevant docs	
non-interpolated	0.1585

Document Level Averages	
	Precision
At 5 docs	0.3440
At 10 docs	0.3240
At 15 docs	0.2987
At 20 docs	0.2810
At 30 docs	0.2600
At 100 docs	0.1726
At 200 docs	0.1217
At 500 docs	0.0706
At 1000 docs	0.0423
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1987

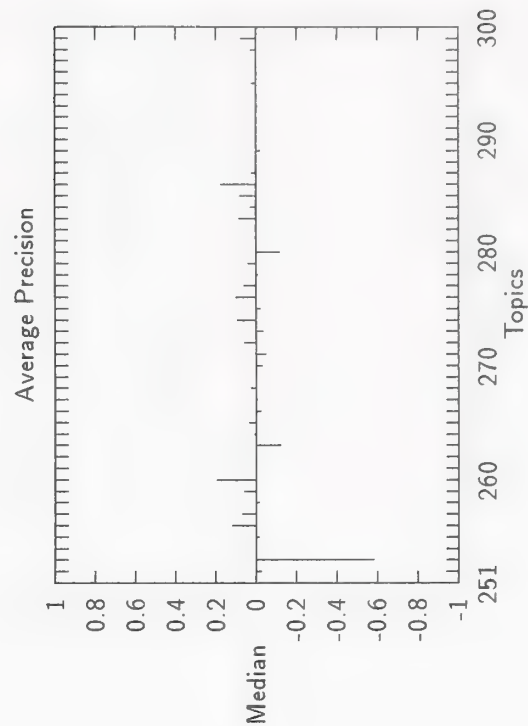
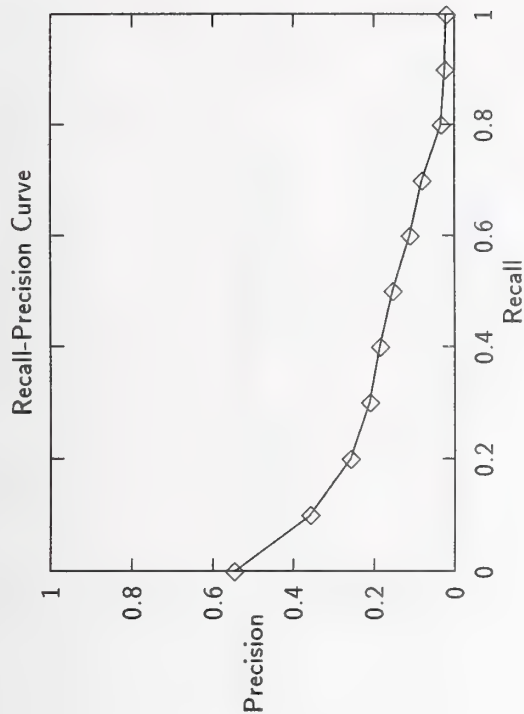


Summary Statistics		
Run Number	ibms96b--category A, automatic, short topic	
Number of Topics		50
Total number of documents over all topics		
Retrieved:		50000
Relevant:		5524
Rel_ret:		2114

Recall Level Precision Averages	
Recall	Precision
0.00	0.5463
0.10	0.3587
0.20	0.2586
0.30	0.2114
0.40	0.1861
0.50	0.1542
0.60	0.1108
0.70	0.0812
0.80	0.0340
0.90	0.0232
1.00	0.0207
Average precision over all relevant docs	
non-interpolated	0.1623

Document Level Averages	
	Precision
At 5 docs	0.3960
At 10 docs	0.3160
At 15 docs	0.2813
At 20 docs	0.2710
At 30 docs	0.2413
At 100 docs	0.1656
At 200 docs	0.1250
At 500 docs	0.0726
At 1000 docs	0.0423

R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2015

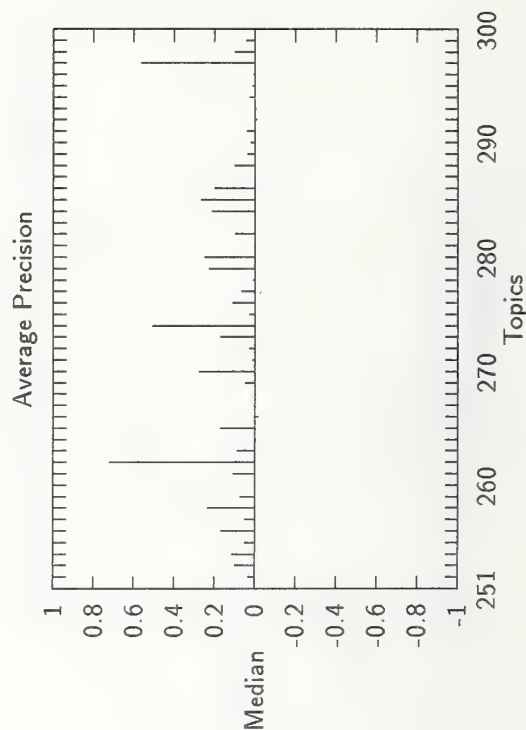
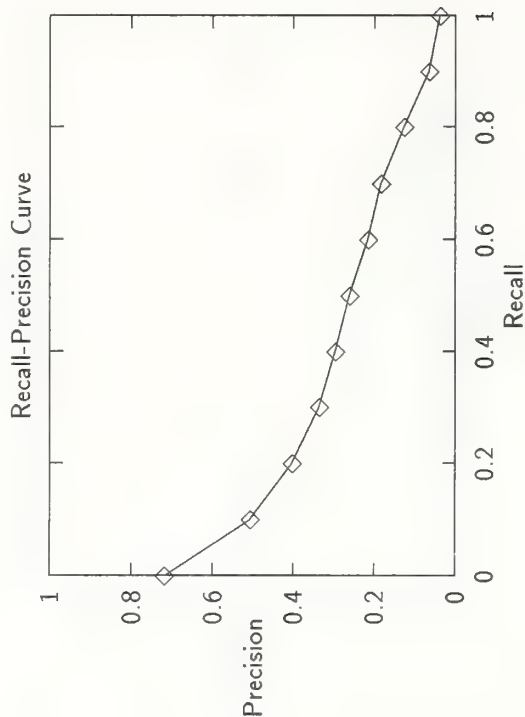


Summary Statistics

Run Number	LNADesc1-category A, manual, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2932

Recall Level Precision Averages	
Recall	Precision
0.00	0.7189
0.10	0.5072
0.20	0.4043
0.30	0.3365
0.40	0.2966
0.50	0.2623
0.60	0.2160
0.70	0.1849
0.80	0.1272
0.90	0.0654
1.00	0.0386
Average precision over all relevant docs	
non-interpolated	0.2661

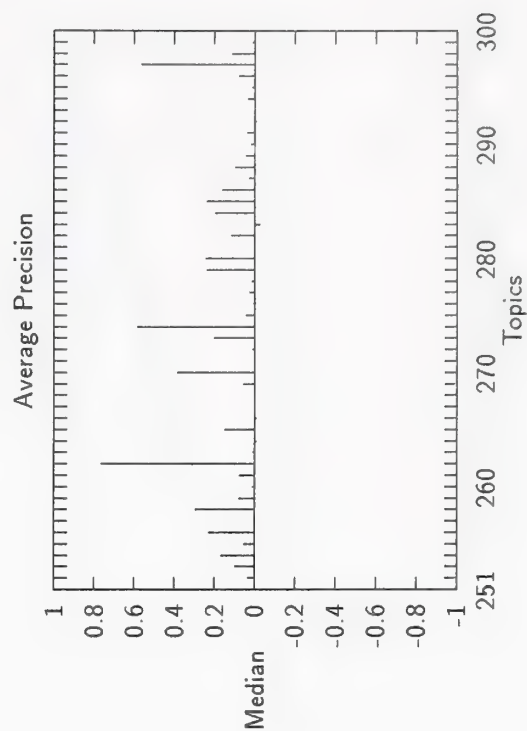
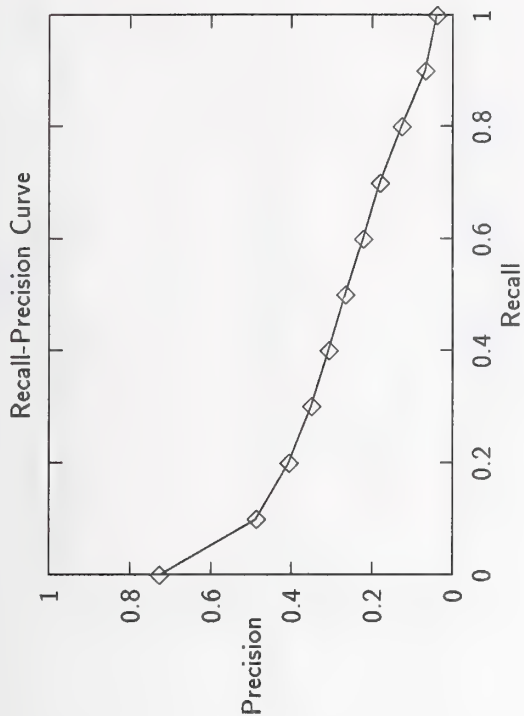
Document Level Averages	
	Precision
At 5 docs	0.5280
At 10 docs	0.4900
At 15 docs	0.4373
At 20 docs	0.4050
At 30 docs	0.3473
At 100 docs	0.2298
At 200 docs	0.1657
At 500 docs	0.0959
At 1000 docs	0.0586
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2887



Summary Statistics		
Run Number	LNaDesc2-category A, manual, short topic	
Number of Topics	50	
Total number of documents over all topics		
Retrieved:	50000	
Relevant:	5524	
Rel_ret:	2929	

Recall Level Precision Averages	
Recall	Precision
0.00	0.7288
0.10	0.4883
0.20	0.4064
0.30	0.3509
0.40	0.3081
0.50	0.2671
0.60	0.2225
0.70	0.1813
0.80	0.1267
0.90	0.0691
1.00	0.0402
Average precision over all relevant docs	
non-interpolated	0.2699

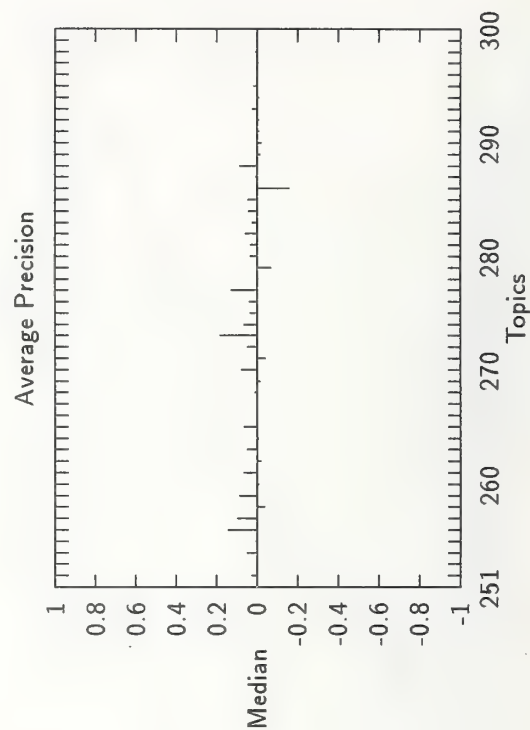
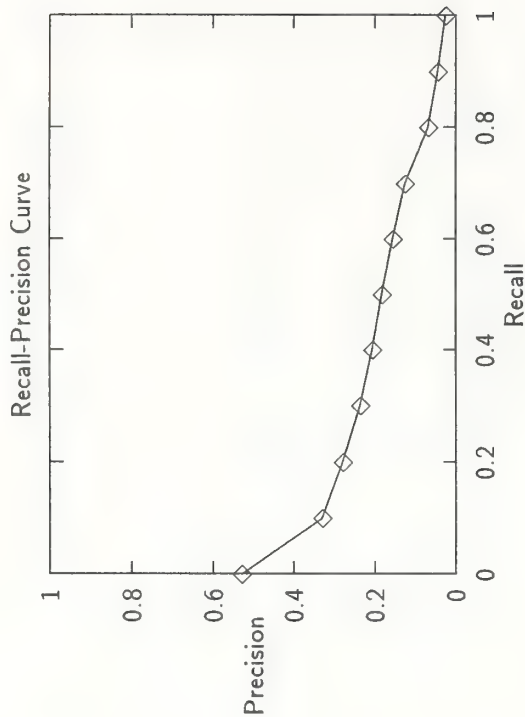
Document Level Averages	
	Precision
At 5 docs	0.5160
At 10 docs	0.4980
At 15 docs	0.4480
At 20 docs	0.4090
At 30 docs	0.3573
At 100 docs	0.2356
At 200 docs	0.1687
At 500 docs	0.0964
At 1000 docs	0.0586
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3043



Summary Statistics	
Run Number	pircsAAS-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2335

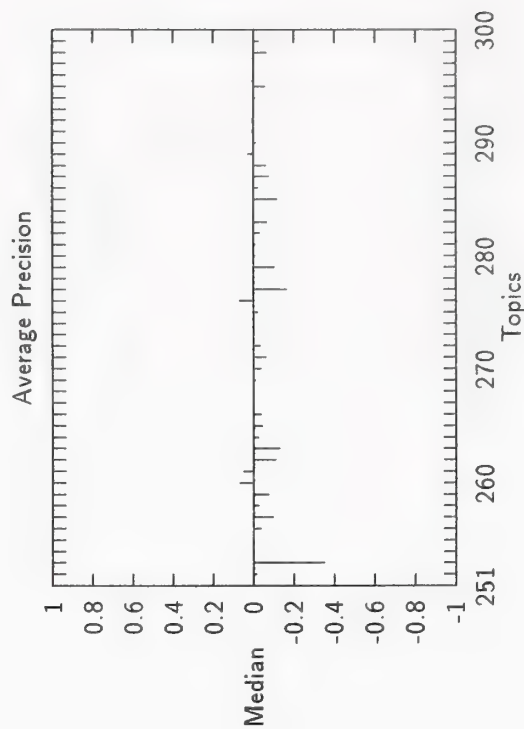
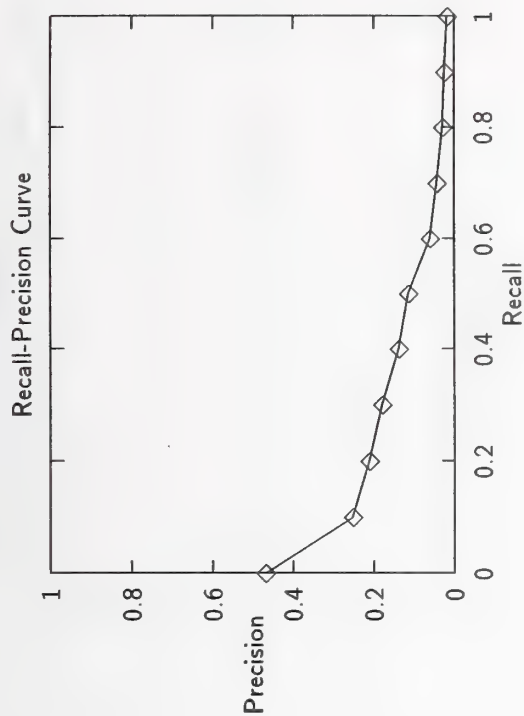
Recall Level Precision Averages	
Recall	Precision
0.00	0.5292
0.10	0.3306
0.20	0.2806
0.30	0.2372
0.40	0.2071
0.50	0.1823
0.60	0.1561
0.70	0.1255
0.80	0.0685
0.90	0.0443
1.00	0.0259
Average precision over all relevant docs	
non-interpolated	0.1809

Document Level Averages	
At 5 docs	0.3720
At 10 docs	0.3260
At 15 docs	0.3040
At 20 docs	0.2760
At 30 docs	0.2453
At 100 docs	0.1728
At 200 docs	0.1285
At 500 docs	0.0759
At 1000 docs	0.0467
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2097



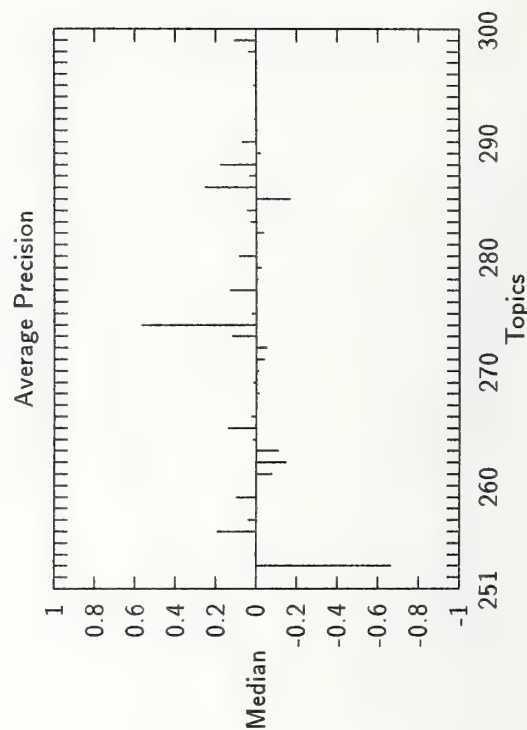
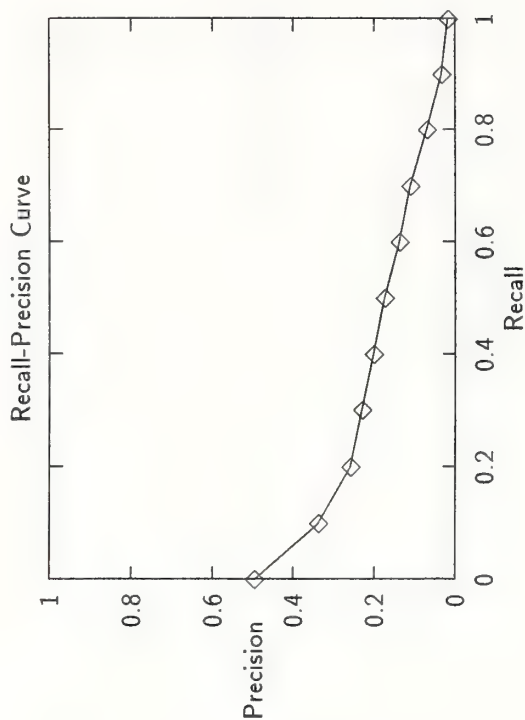
Summary Statistics	
Run Number	mds002-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	1867

Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.4685	At 5 docs	0.2680
0.10	0.2526	At 10 docs	0.2320
0.20	0.2111	At 15 docs	0.2227
0.30	0.1797	At 20 docs	0.2160
0.40	0.1383	At 30 docs	0.2013
0.50	0.1150	At 100 docs	0.1368
0.60	0.0607	At 200 docs	0.1044
0.70	0.0423	At 500 docs	0.0617
0.80	0.0283	At 1000 docs	0.0373
0.90	0.0236	R—Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0182		
Average precision over all relevant docs		Exact	0.1607
non-interpolated	0.1214		



Summary Statistics	
Run Number	ETHas1-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2514

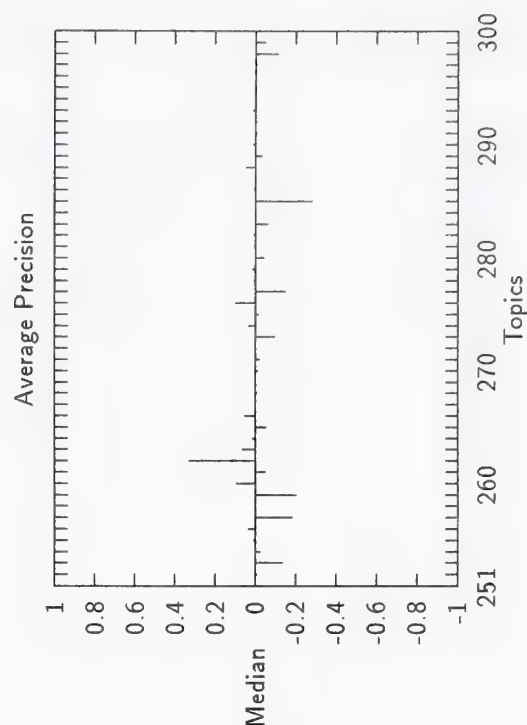
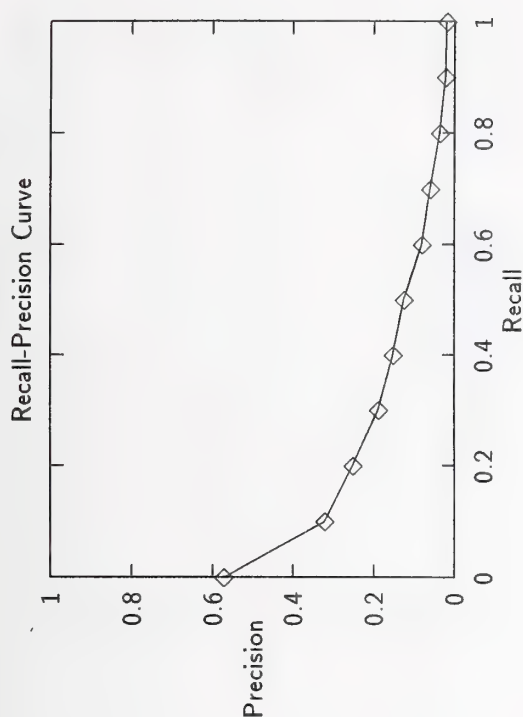
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.4959	At 5 docs	0.3240
0.10	0.3381	At 10 docs	0.3060
0.20	0.2598	At 15 docs	0.3013
0.30	0.2309	At 20 docs	0.2840
0.40	0.2025	At 30 docs	0.2593
0.50	0.1749	At 100 docs	0.1782
0.60	0.1380	At 200 docs	0.1325
0.70	0.1105	At 500 docs	0.0798
0.80	0.0701	At 1000 docs	0.0503
0.90	0.0338	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0168		
Average precision over all relevant docs		Exact	0.1977
non-interpolated	0.1726		



Summary Statistics	
Run Number	brkly15-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	1958

Recall Level Precision Averages	
Recall	Precision
0.00	0.5716
0.10	0.3228
0.20	0.2527
0.30	0.1892
0.40	0.1533
0.50	0.1257
0.60	0.0808
0.70	0.0600
0.80	0.0358
0.90	0.0225
1.00	0.0187
Average precision over all relevant docs	
non-interpolated	0.1420

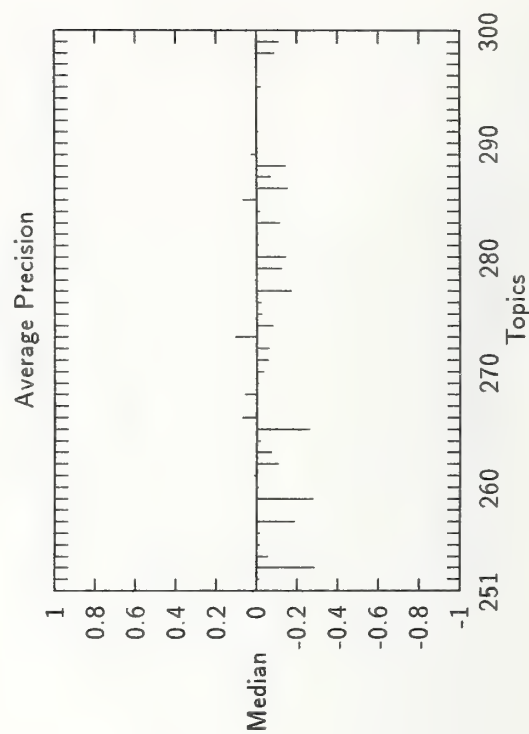
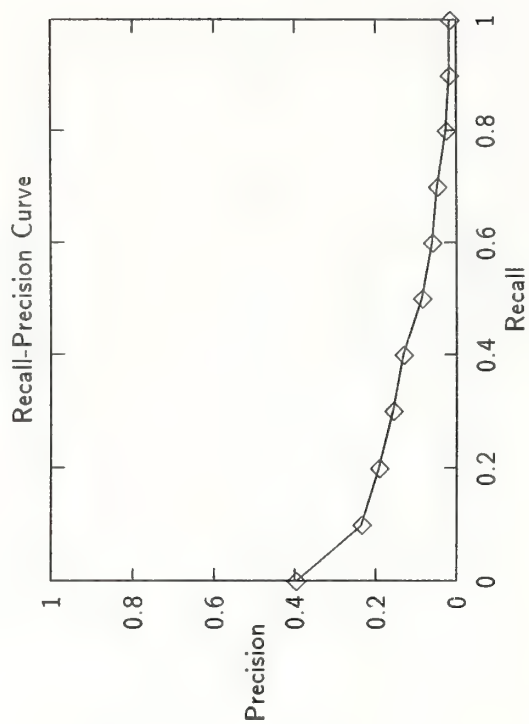
Document Level Averages	
At 5 docs	0.3560
At 10 docs	0.3060
At 15 docs	0.2693
At 20 docs	0.2520
At 30 docs	0.2200
At 100 docs	0.1416
At 200 docs	0.1033
At 500 docs	0.0596
At 1000 docs	0.0392
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1788



Summary Statistics	
Run Number	KUSG2-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2011

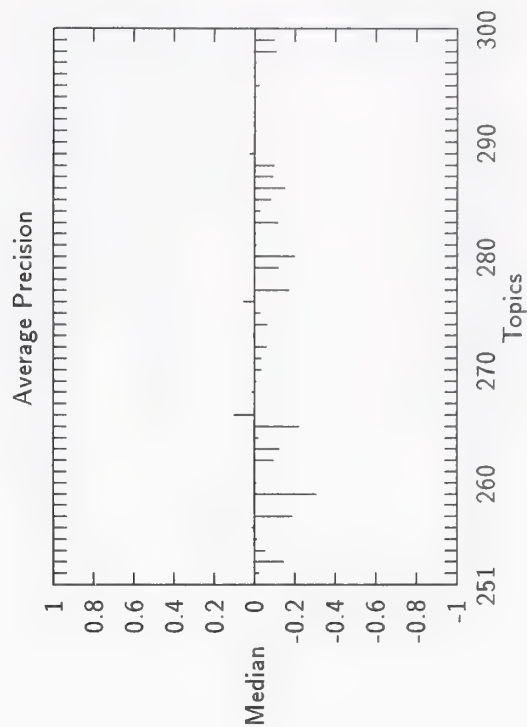
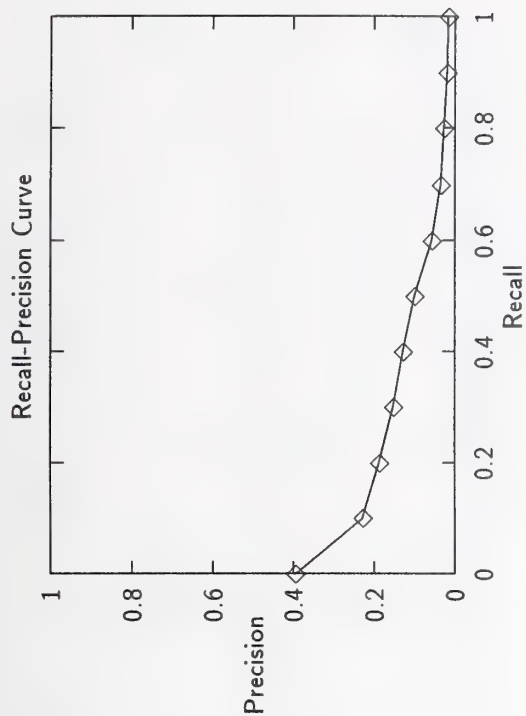
Recall Level Precision Averages	
Recall	Precision
0.00	0.3971
0.10	0.2354
0.20	0.1931
0.30	0.1566
0.40	0.1316
0.50	0.0857
0.60	0.0610
0.70	0.0473
0.80	0.0254
0.90	0.0182
1.00	0.0158
Average precision over all relevant docs	
non-interpolated	0.1073

Document Level Averages	
At 5 docs	0.2240
At 10 docs	0.1960
At 15 docs	0.1973
At 20 docs	0.1900
At 30 docs	0.1747
At 100 docs	0.1260
At 200 docs	0.0987
At 500 docs	0.0624
At 1000 docs	0.0402
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1526



Summary Statistics	
Run Number	KUSG3—category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
RelRet:	1926

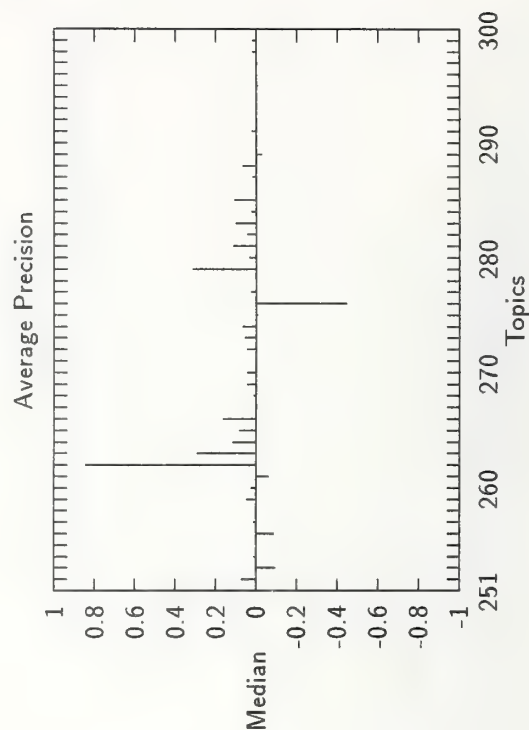
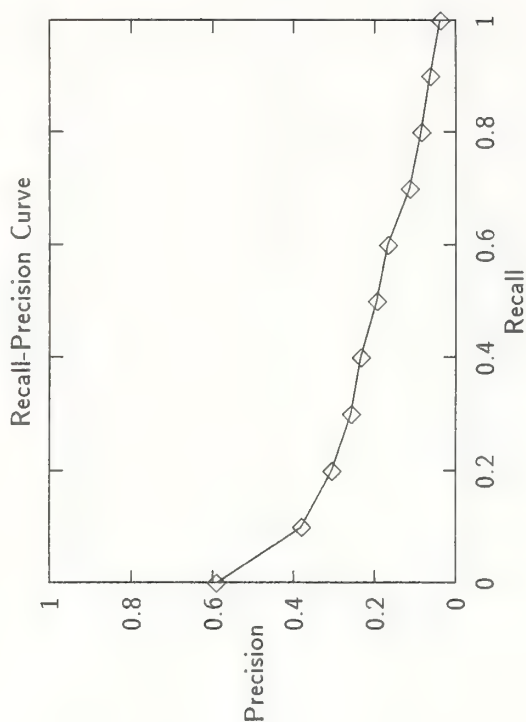
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.3956	At 5 docs	0.2160
0.10	0.2294	At 10 docs	0.2020
0.20	0.1875	At 15 docs	0.1907
0.30	0.1527	At 20 docs	0.1860
0.40	0.1285	At 30 docs	0.1753
0.50	0.1005	At 100 docs	0.1246
0.60	0.0567	At 200 docs	0.0945
0.70	0.0357	At 500 docs	0.0598
0.80	0.0269	At 1000 docs	0.0385
0.90	0.0187	R—Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0163		
Average precision over all relevant docs		Exact	0.1482
non-interpolated	0.1056		



Summary Statistics	
Run Number	INQ301-category A, automatic, short topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel.Ret:	2817

Recall Level Precision Averages	
Recall	Precision
0.00	0.5919
0.10	0.3811
0.20	0.3071
0.30	0.2586
0.40	0.2359
0.50	0.1955
0.60	0.1682
0.70	0.1138
0.80	0.0856
0.90	0.0643
1.00	0.0378
Average precision over all relevant docs	
non-interpolated	0.2002

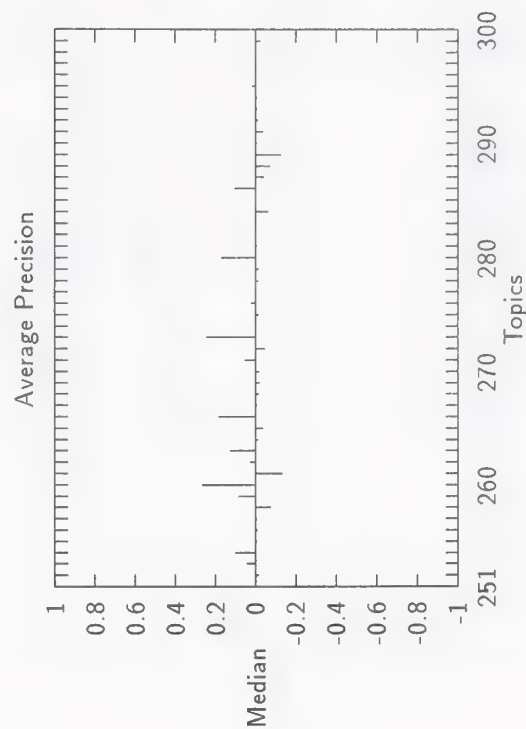
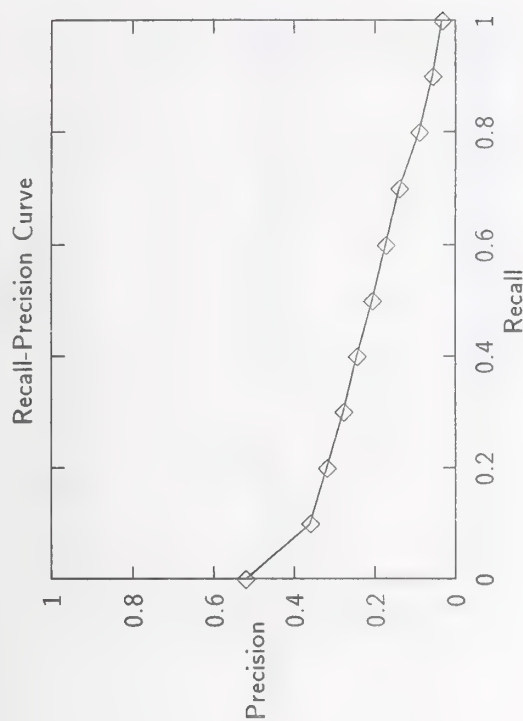
Document Level Averages	
At 5 docs	0.3760
At 10 docs	0.3320
At 15 docs	0.3080
At 20 docs	0.2970
At 30 docs	0.2767
At 100 docs	0.1736
At 200 docs	0.1362
At 500 docs	0.0896
At 1000 docs	0.0563
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2347



Summary Statistics	
Run Number	vtwnB1-category A, automatic, long topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2532

Recall Level Precision Averages	
Recall	Precision
0.00	0.5216
0.10	0.3611
0.20	0.3199
0.30	0.2785
0.40	0.2442
0.50	0.2061
0.60	0.1739
0.70	0.1393
0.80	0.0914
0.90	0.0568
1.00	0.0327
Average precision over all relevant docs	
non-interpolated	0.2065

Document Level Averages	
	Precision
At 5 docs	0.3640
At 10 docs	0.3240
At 15 docs	0.3027
At 20 docs	0.2780
At 30 docs	0.2593
At 100 docs	0.1724
At 200 docs	0.1317
At 500 docs	0.0812
At 1000 docs	0.0506
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2268



Summary Statistics

Run Number	city96a1-category A, automatic, long topic
Number of Topics	50

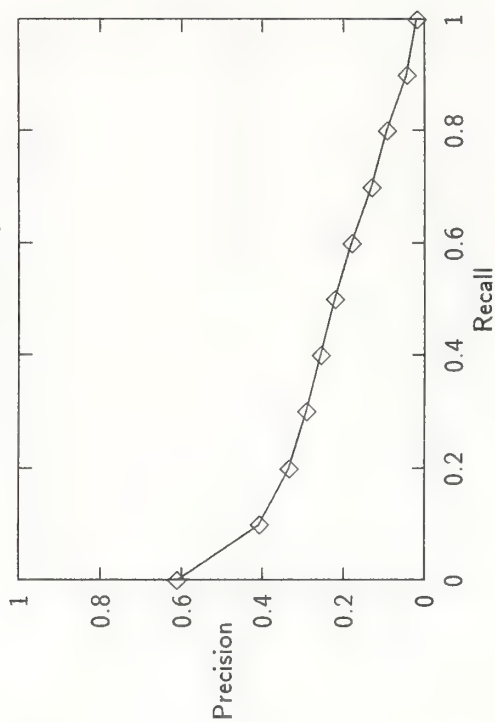
Total number of documents over all topics

Retrieved:	50000
Relevant:	5524
Rel-ret:	2960

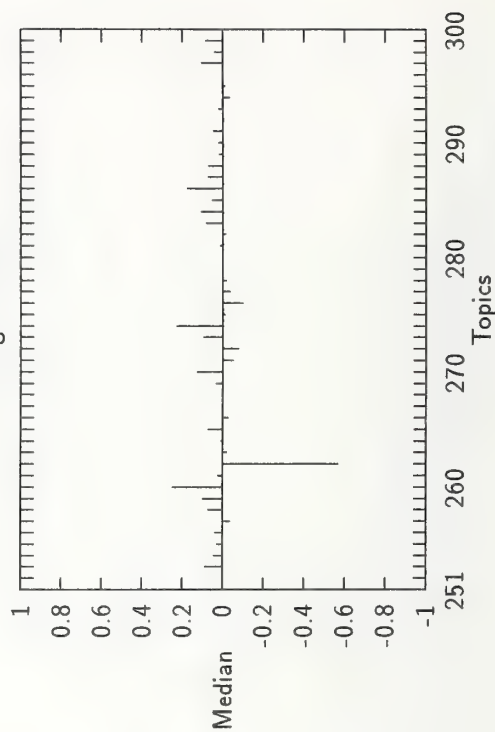
Recall Level Precision Averages	
Recall	Precision
0.00	0.6121
0.10	0.4091
0.20	0.3347
0.30	0.2924
0.40	0.2571
0.50	0.2219
0.60	0.1814
0.70	0.1327
0.80	0.0943
0.90	0.0444
1.00	0.0193
Average precision over all relevant docs	
non-interpolated	0.2163

Document Level Averages	
At 5 docs	0.4280
At 10 docs	0.3900
At 15 docs	0.3733
At 20 docs	0.3580
At 30 docs	0.3240
At 100 docs	0.2160
At 200 docs	0.1600
At 500 docs	0.0957
At 1000 docs	0.0592
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2478

Recall-Precision Curve



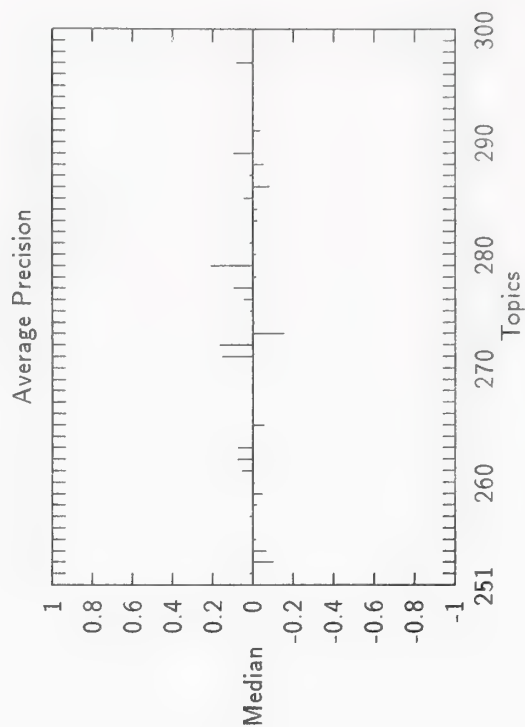
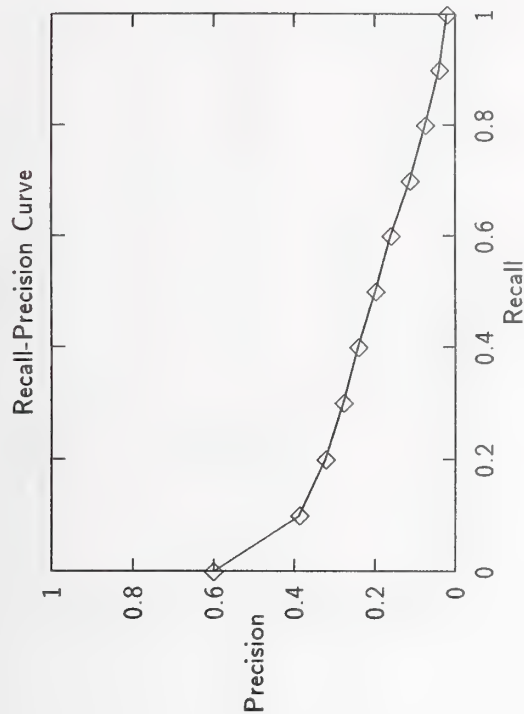
Average Precision



Summary Statistics	
Run Number	genrl2-category A, automatic, long topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel.ret:	2501

Recall Level Precision Averages	
Recall	Precision
0.00	0.6008
0.10	0.3877
0.20	0.3220
0.30	0.2790
0.40	0.2411
0.50	0.1991
0.60	0.1610
0.70	0.1130
0.80	0.0740
0.90	0.0403
1.00	0.0206
Average precision over all relevant docs	
non-interpolated	0.2024

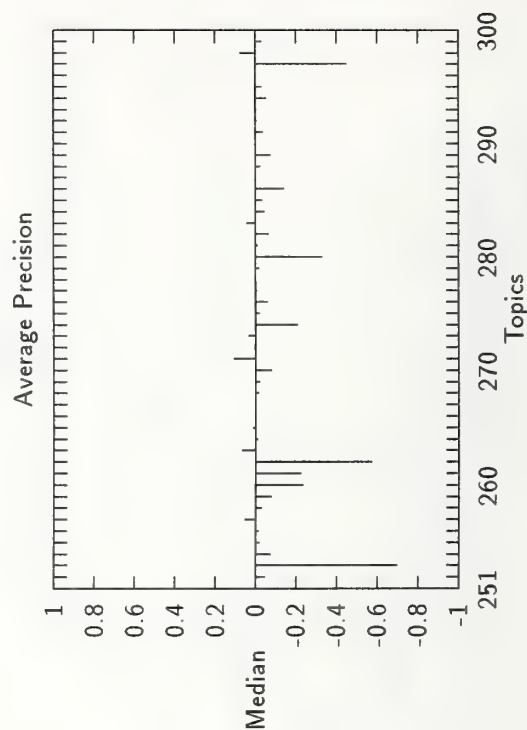
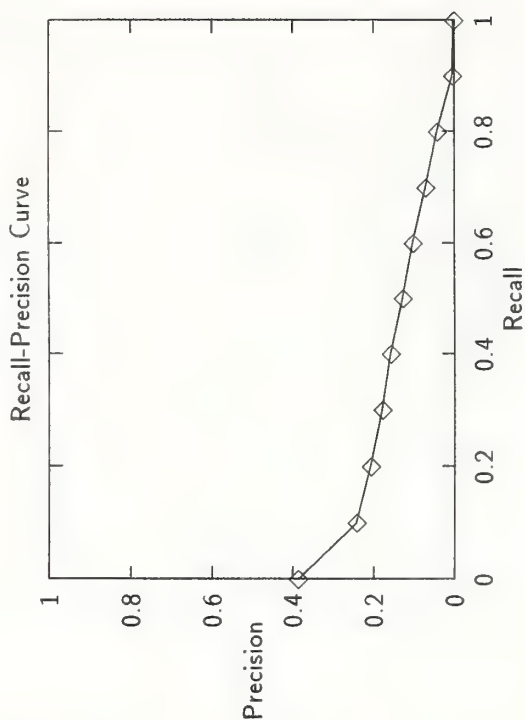
Document Level Averages	
At 5 docs	0.4440
At 10 docs	0.3600
At 15 docs	0.3267
At 20 docs	0.3070
At 30 docs	0.2827
At 100 docs	0.1830
At 200 docs	0.1327
At 500 docs	0.0804
At 1000 docs	0.0500
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2395



Summary Statistics	
Run Number	gmu96au2-category A, automatic, long topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel.Ret:	2082

Recall Level Precision Averages	
Recall	Precision
0.00	0.3875
0.10	0.2426
0.20	0.2089
0.30	0.1802
0.40	0.1587
0.50	0.1288
0.60	0.1039
0.70	0.0719
0.80	0.0431
0.90	0.0042
1.00	0.0011
Average precision over all relevant docs	
non-interpolated	0.1256

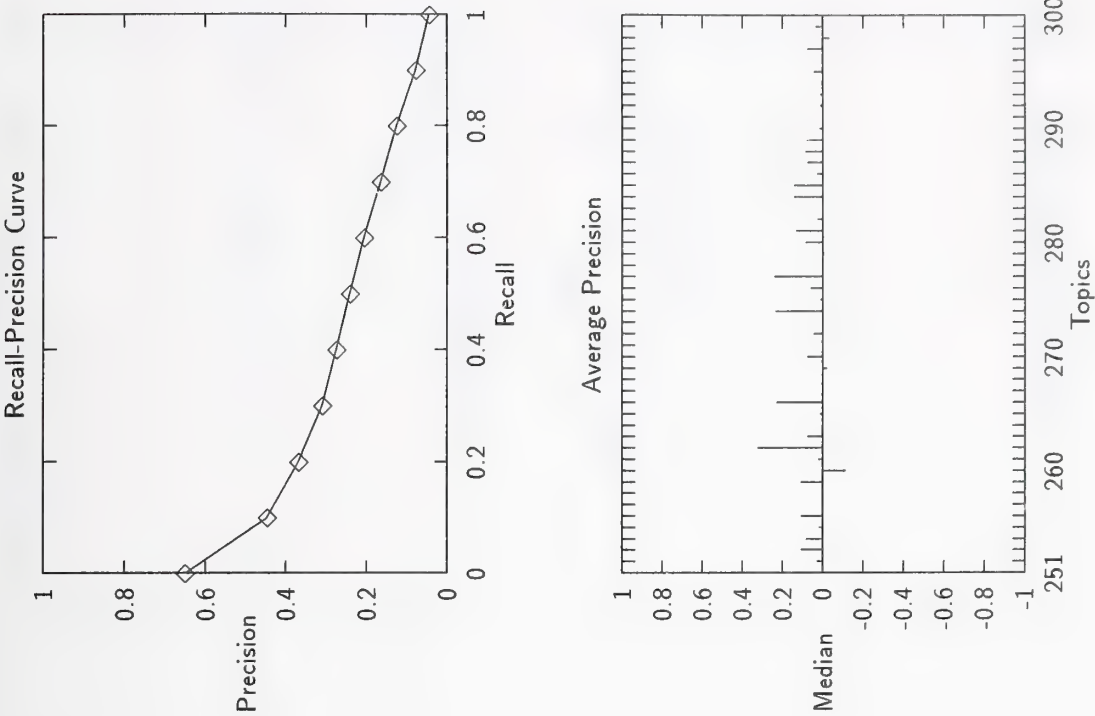
Document Level Averages	
	Precision
At 5 docs	0.2200
At 10 docs	0.2160
At 15 docs	0.2133
At 20 docs	0.2060
At 30 docs	0.1873
At 100 docs	0.1360
At 200 docs	0.1043
At 500 docs	0.0658
At 1000 docs	0.0416
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1613



Summary Statistics		
Run Number	pircsAAL-category A, automatic, long topic	50
Number of Topics		
Total number of documents over all topics		
Retrieved:	50000	
Relevant:	5524	
Rel_ret:	2941	

Recall Level Precision Averages	
Recall	Precision
0.00	0.6496
0.10	0.4474
0.20	0.3691
0.30	0.3096
0.40	0.2746
0.50	0.2412
0.60	0.2068
0.70	0.1655
0.80	0.1257
0.90	0.0775
1.00	0.0439
Average precision over all relevant docs	
non-interpolated	0.2466

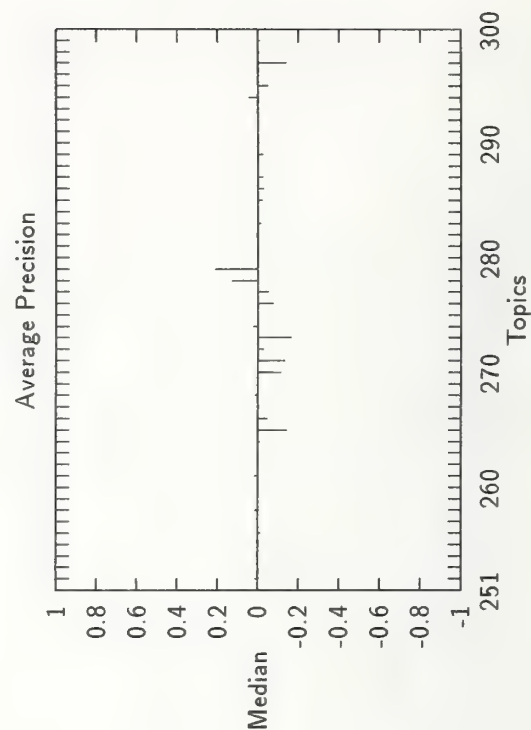
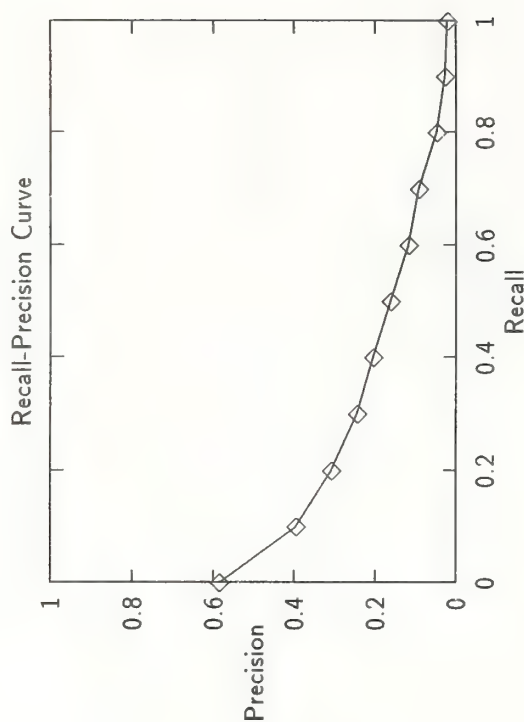
Document Level Averages	
At 5 docs	0.4640
At 10 docs	0.4260
At 15 docs	0.3933
At 20 docs	0.3650
At 30 docs	0.3247
At 100 docs	0.2138
At 200 docs	0.1541
At 500 docs	0.0954
At 1000 docs	0.0588
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2672



Summary Statistics	
Run Number	mds001-category A, automatic, long topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel.Ret:	2375

Recall Level Precision Averages	
Recall	Precision
0.00	0.5856
0.10	0.3956
0.20	0.3083
0.30	0.2455
0.40	0.2057
0.50	0.1623
0.60	0.1183
0.70	0.0930
0.80	0.0481
0.90	0.0263
1.00	0.0207
Average precision over all relevant docs	
non-interpolated	0.1804

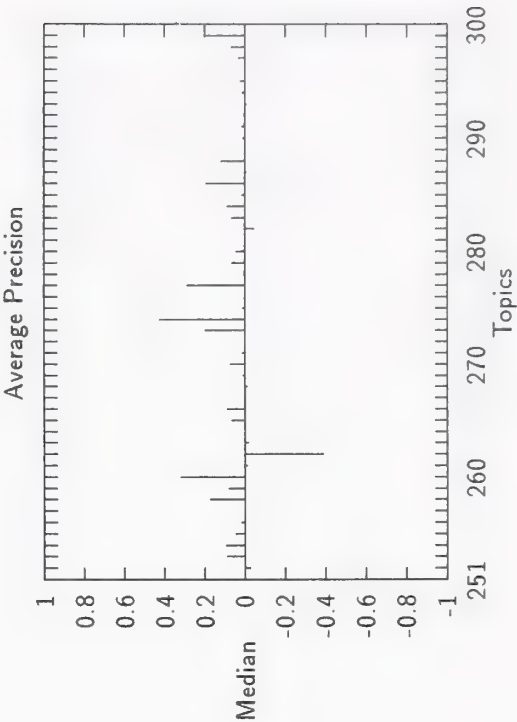
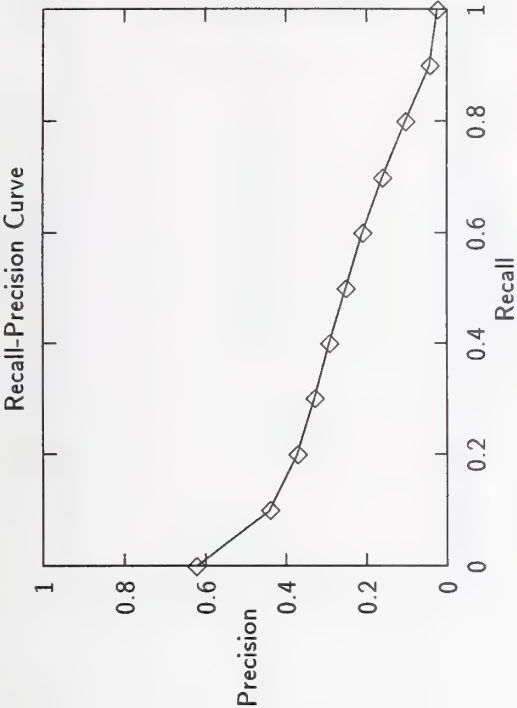
Document Level Averages	
	Precision
At 5 docs	0.4040
At 10 docs	0.3680
At 15 docs	0.3373
At 20 docs	0.3050
At 30 docs	0.2760
At 100 docs	0.1650
At 200 docs	0.1221
At 500 docs	0.0744
At 1000 docs	0.0475
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2292



Summary Statistics	
Run Number	ETHall-category A, automatic, long topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2989

Recall Level Precision Averages	
Recall	Precision
0.00	0.6231
0.10	0.4403
0.20	0.3718
0.30	0.3293
0.40	0.2922
0.50	0.2519
0.60	0.2093
0.70	0.1599
0.80	0.1039
0.90	0.0435
1.00	0.0236
Average precision over all relevant docs	
non-interpolated	0.2425

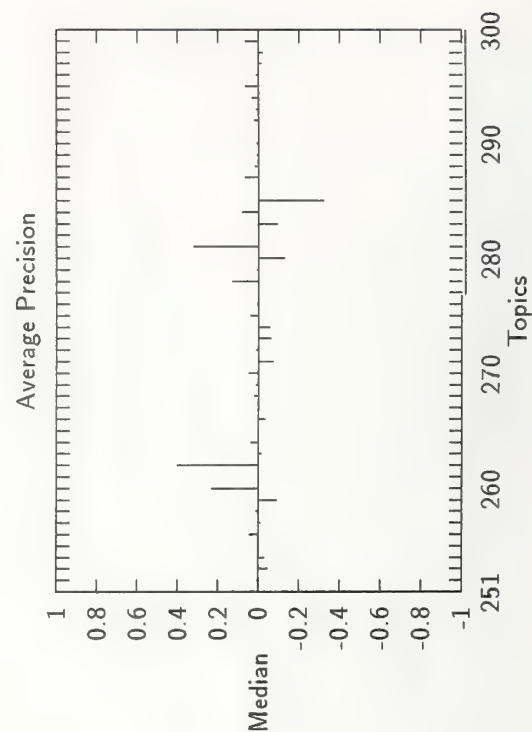
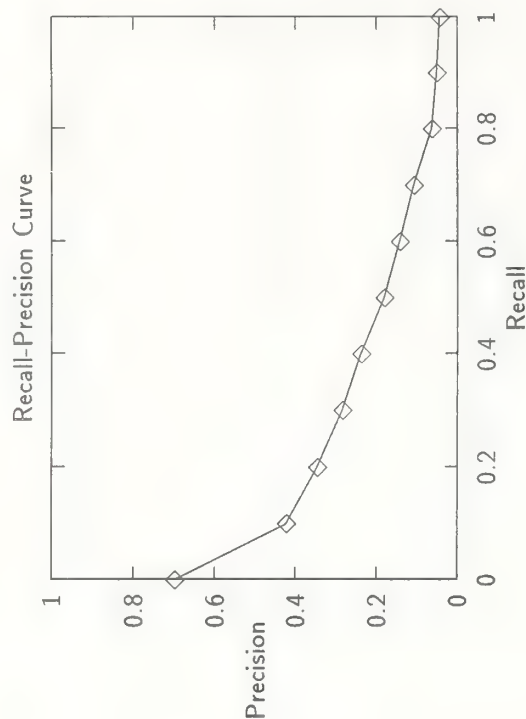
Document Level Averages	
	Precision
At 5 docs	0.4360
At 10 docs	0.4260
At 15 docs	0.4013
At 20 docs	0.3720
At 30 docs	0.3327
At 100 docs	0.2206
At 200 docs	0.1637
At 500 docs	0.0966
At 1000 docs	0.0598
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2605



Summary Statistics	
Run Number	brkly16-category A, automatic, long topic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel _{ret} :	2472

Recall Level Precision Averages	
Recall	Precision
0.00	0.6982
0.10	0.4231
0.20	0.3467
0.30	0.2846
0.40	0.2389
0.50	0.1812
0.60	0.1428
0.70	0.1080
0.80	0.0641
0.90	0.0506
1.00	0.0426
Average precision over all relevant docs	
non-interpolated	0.2076

Document Level Averages	
	Precision
At 5 docs	0.4640
At 10 docs	0.3880
At 15 docs	0.3520
At 20 docs	0.3260
At 30 docs	0.2967
At 100 docs	0.1878
At 200 docs	0.1366
At 500 docs	0.0791
At 1000 docs	0.0494
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2406

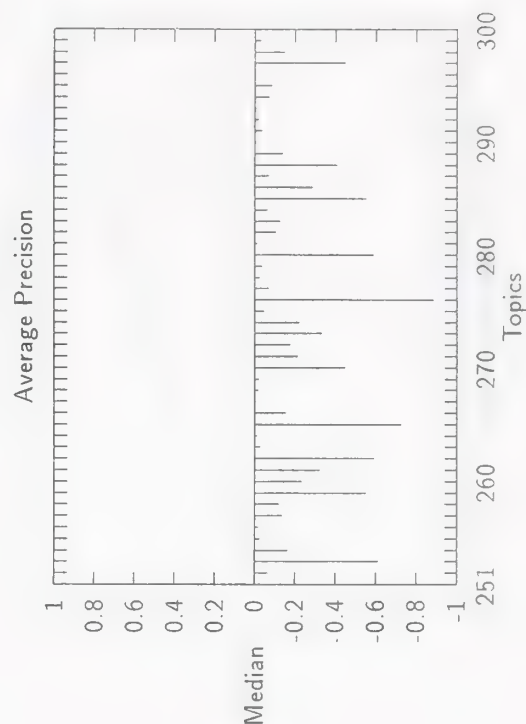
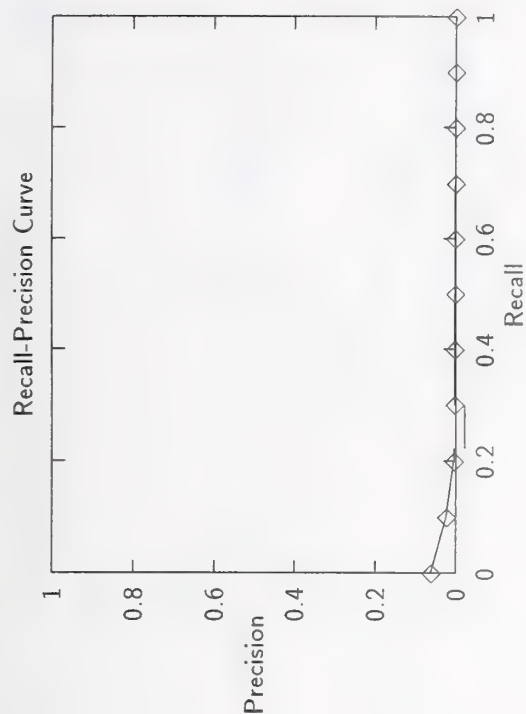


Summary Statistics		
Run Number	INQ302-category A, automatic, long topic	50
Number of Topics		
Total number of documents over all topics		
Retrieved:	50000	
Relevant:	5524	
Rel_ret:	202	

Recall Level Precision Averages	
Recall	Precision
0.00	0.0619
0.10	0.0244
0.20	0.0041
0.30	0.0039
0.40	0.0037
0.50	0.0012
0.60	0.0008
0.70	0.0006
0.80	0.0000
0.90	0.0000
1.00	0.0000

Average precision over all relevant docs	
non-interpolated	0.0043

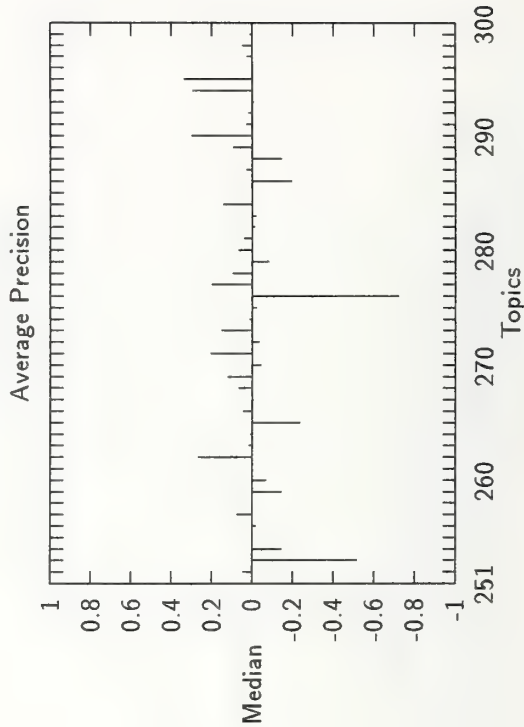
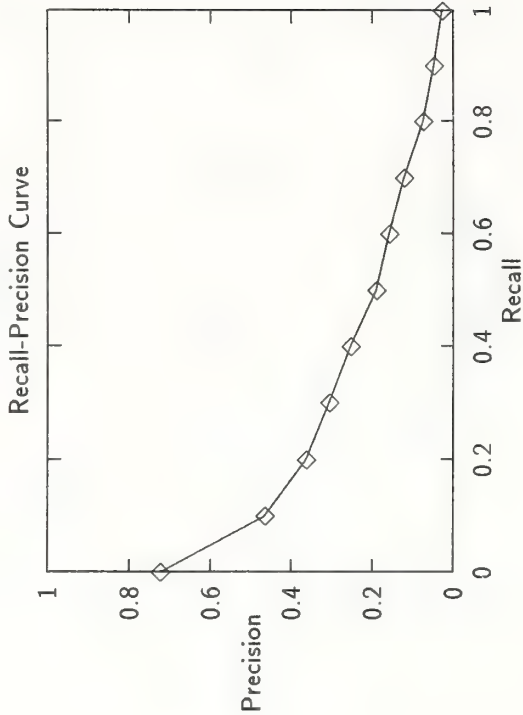
Document Level Averages	
	Precision
At 5 docs	0.0160
At 10 docs	0.0100
At 15 docs	0.0080
At 20 docs	0.0060
At 30 docs	0.0047
At 100 docs	0.0026
At 200 docs	0.0021
At 500 docs	0.0034
At 1000 docs	0.0040
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0054



Summary Statistics	
Run Number	anu5man4-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	33220
Relevant:	5524
Rel_ret:	2889

Recall Level Precision Averages	
Recall	Precision
0.00	0.7231
0.10	0.4662
0.20	0.3644
0.30	0.3070
0.40	0.2549
0.50	0.1905
0.60	0.1583
0.70	0.1217
0.80	0.0751
0.90	0.0483
1.00	0.0273
Average precision over all relevant docs	
non-interpolated	0.2261

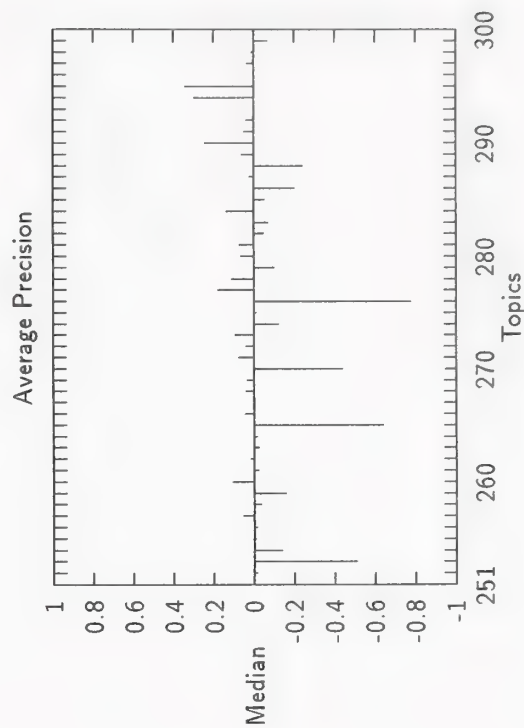
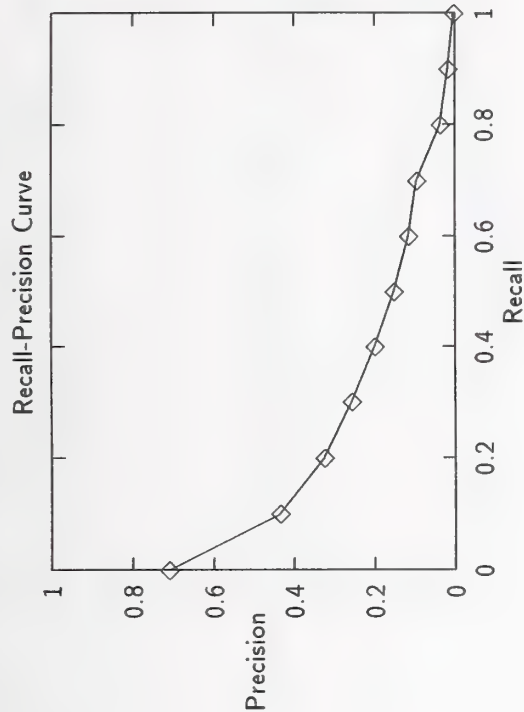
Document Level Averages	
At 5 docs	0.4920
At 10 docs	0.4540
At 15 docs	0.4200
At 20 docs	0.3840
At 30 docs	0.3520
At 100 docs	0.2456
At 200 docs	0.1731
At 500 docs	0.0973
At 1000 docs	0.0578
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2689



Summary Statistics	
Run Number	anu5man6-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	30337
Relevant:	5524
Rel_ret:	2405

Recall Level Precision Averages	
Recall	Precision
0.00	0.7103
0.10	0.4353
0.20	0.3262
0.30	0.2569
0.40	0.2000
0.50	0.1520
0.60	0.1163
0.70	0.0956
0.80	0.0358
0.90	0.0176
1.00	0.0036
Average precision over all relevant docs	
non-interpolated	0.1889

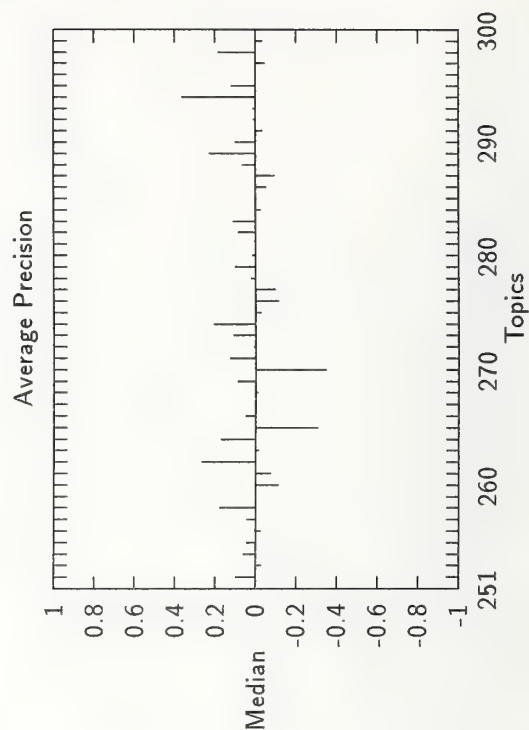
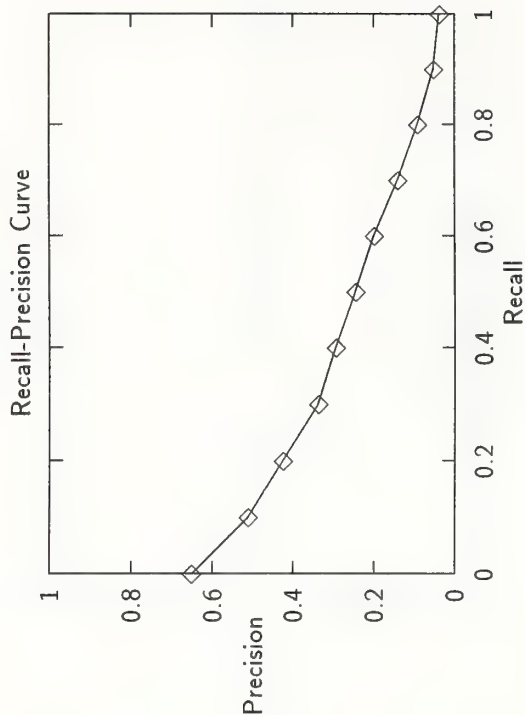
Document Level Averages	
	Precision
At 5 docs	0.4680
At 10 docs	0.4120
At 15 docs	0.3800
At 20 docs	0.3580
At 30 docs	0.3073
At 100 docs	0.2082
At 200 docs	0.1419
At 500 docs	0.0801
At 1000 docs	0.0481
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2313



Summary Statistics	
Run Number	CLCLUS-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	3163

Recall Level Precision Averages	
Recall	Precision
0.00	0.6525
0.10	0.5120
0.20	0.4267
0.30	0.3393
0.40	0.2955
0.50	0.2458
0.60	0.1997
0.70	0.1409
0.80	0.0924
0.90	0.0518
1.00	0.0378
Average precision over all relevant docs	
non-interpolated	0.2535

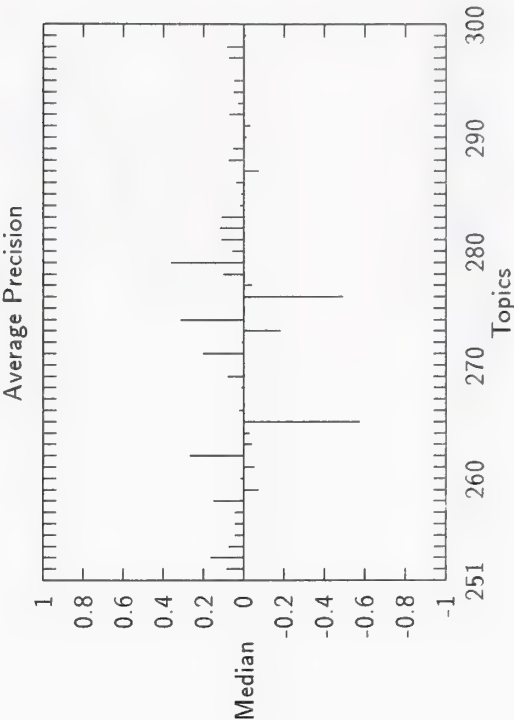
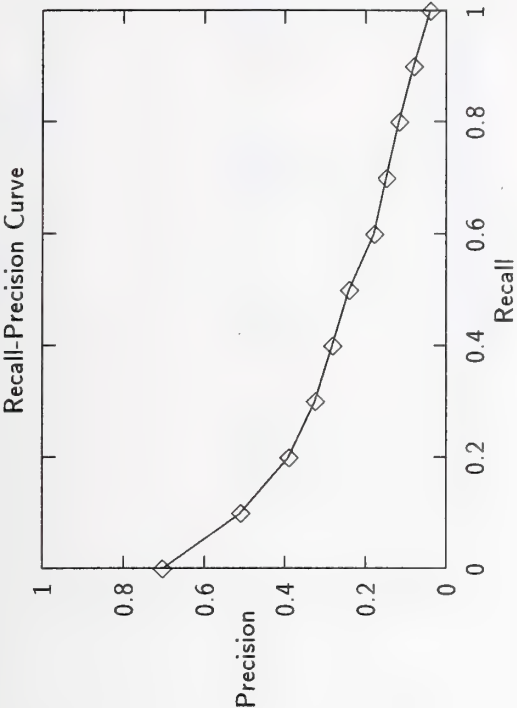
Document Level Averages	
At 5 docs	0.4480
At 10 docs	0.4380
At 15 docs	0.4360
At 20 docs	0.4140
At 30 docs	0.3680
At 100 docs	0.2688
At 200 docs	0.1874
At 500 docs	0.1046
At 1000 docs	0.0633
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3085



Summary Statistics	
Run Number	CLTHES-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel.Ret:	3147

Recall Level Precision Averages	
Recall	Precision
0.00	0.7046
0.10	0.5109
0.20	0.3904
0.30	0.3261
0.40	0.2823
0.50	0.2413
0.60	0.1803
0.70	0.1502
0.80	0.1179
0.90	0.0804
1.00	0.0395
Average precision over all relevant docs	
non-interpolated	0.2513

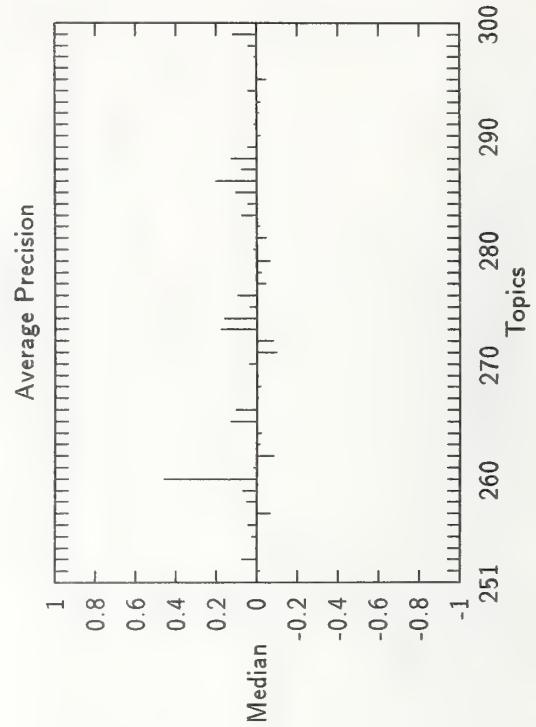
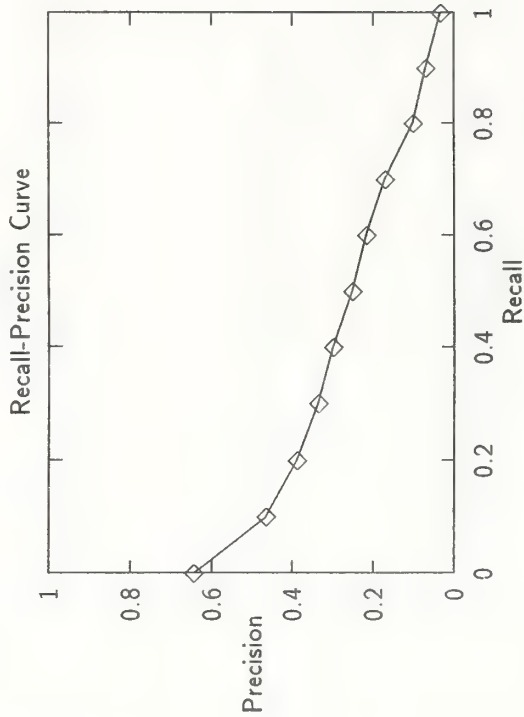
Document Level Averages	
At 5 docs	0.4440
At 10 docs	0.4280
At 15 docs	0.4000
At 20 docs	0.3780
At 30 docs	0.3433
At 100 docs	0.2330
At 200 docs	0.1624
At 500 docs	0.0990
At 1000 docs	0.0629
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2833



Summary Statistics	
Run Number	Cor5M11e-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	3092

Recall Level Precision Averages	
Recall	Precision
0.00	0.6436
0.10	0.4658
0.20	0.3903
0.30	0.3374
0.40	0.2998
0.50	0.2526
0.60	0.2174
0.70	0.1715
0.80	0.1022
0.90	0.0697
1.00	0.0340
Average precision over all relevant docs	
non-interpolated	0.2544

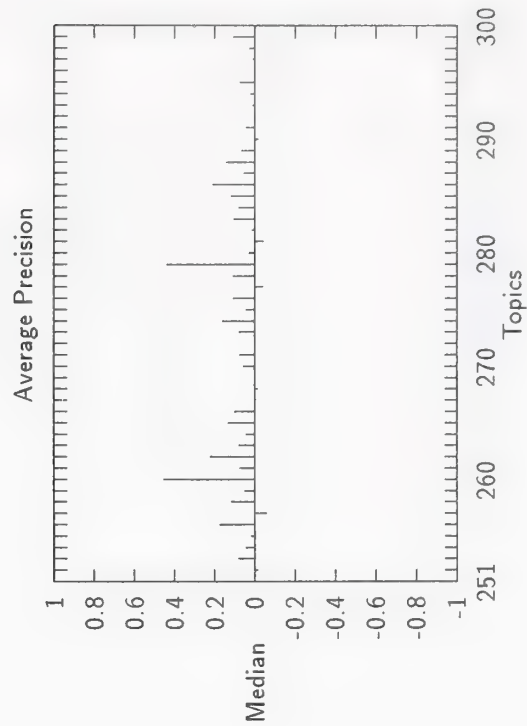
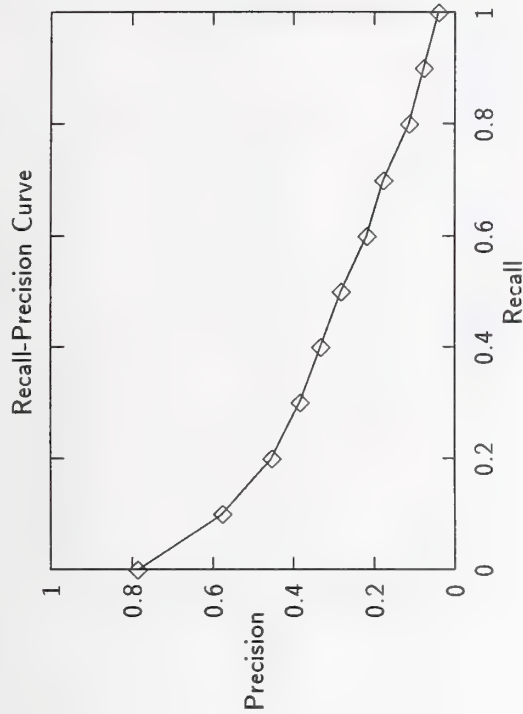
Document Level Averages	
At 5 docs	0.4960
At 10 docs	0.4480
At 15 docs	0.4173
At 20 docs	0.3910
At 30 docs	0.3433
At 100 docs	0.2272
At 200 docs	0.1689
At 500 docs	0.1008
At 1000 docs	0.0618
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2845



Summary Statistics	
Run Number	Cor5M2rf-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	3085

Recall Level Precision Averages	
Recall	Precision
0.00	0.7877
0.10	0.5768
0.20	0.4540
0.30	0.3850
0.40	0.3335
0.50	0.2831
0.60	0.2197
0.70	0.1773
0.80	0.1144
0.90	0.0784
1.00	0.0416
Average precision over all relevant docs	
non-interpolated	0.2931

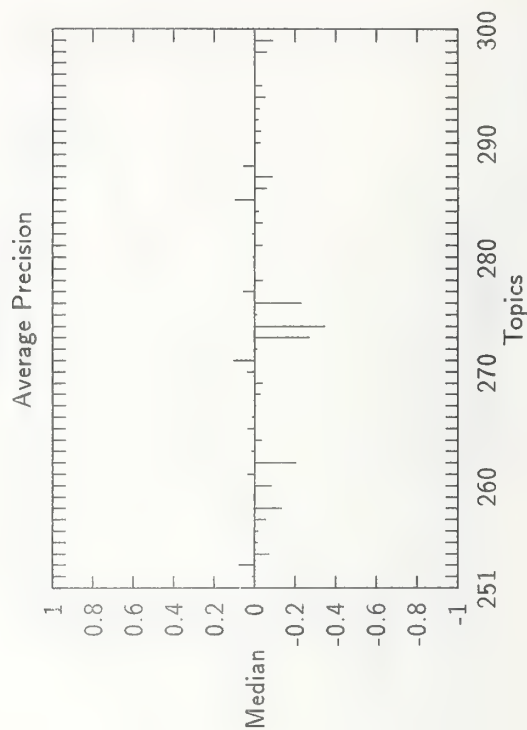
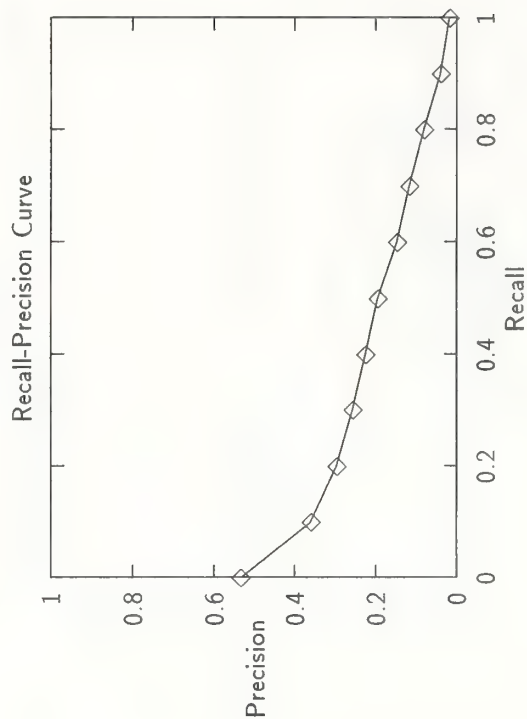
Document Level Averages	
	Precision
At 5 docs	0.5960
At 10 docs	0.5280
At 15 docs	0.4747
At 20 docs	0.4350
At 30 docs	0.3767
At 100 docs	0.2412
At 200 docs	0.1757
At 500 docs	0.1033
At 1000 docs	0.0617
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3112



Summary Statistics	
Run Number	DCU961-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel.ret:	2384

Recall Level Precision Averages	
Recall	Precision
0.00	0.5336
0.10	0.3615
0.20	0.2979
0.30	0.2571
0.40	0.2254
0.50	0.1955
0.60	0.1478
0.70	0.1173
0.80	0.0814
0.90	0.0403
1.00	0.0181
Average precision over all relevant docs	
non-interpolated	0.1862

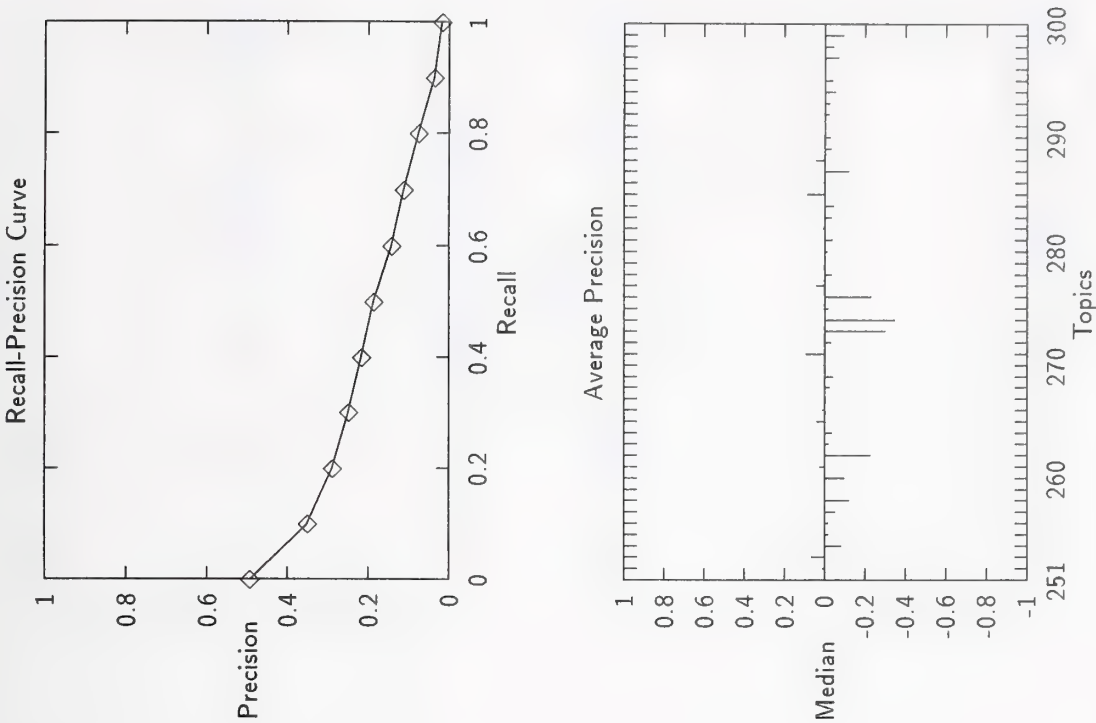
Document Level Averages	
At 5 docs	0.3600
At 10 docs	0.3320
At 15 docs	0.3080
At 20 docs	0.2820
At 30 docs	0.2547
At 100 docs	0.1678
At 200 docs	0.1246
At 500 docs	0.0764
At 1000 docs	0.0477
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2161



Summary Statistics	
Run Number	DCU963—category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2472

Recall Level Precision Averages	
Recall	Precision
0.00	0.4952
0.10	0.3507
0.20	0.2897
0.30	0.2492
0.40	0.2171
0.50	0.1883
0.60	0.1437
0.70	0.1136
0.80	0.0756
0.90	0.0367
1.00	0.0177
Average precision over all relevant docs	
non-interpolated	0.1804

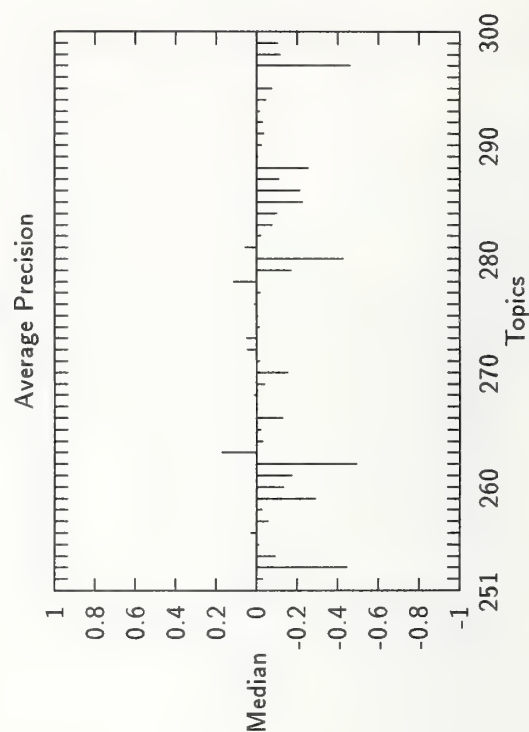
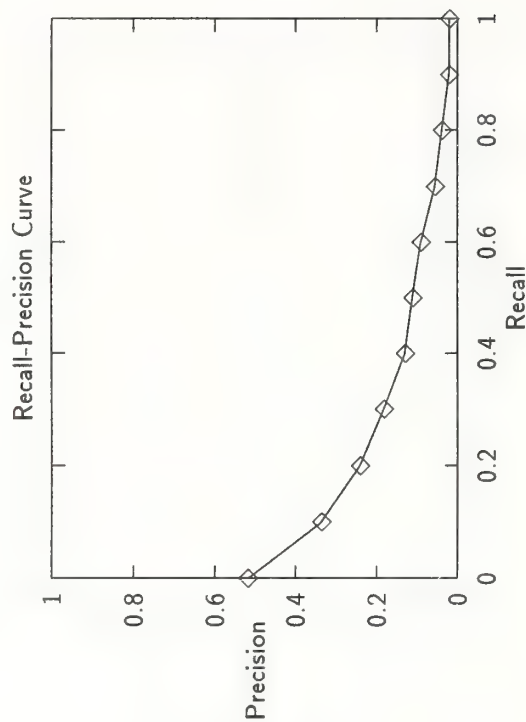
Document Level Averages	
At 5 docs	0.3360
At 10 docs	0.3160
At 15 docs	0.2947
At 20 docs	0.2750
At 30 docs	0.2427
At 100 docs	0.1660
At 200 docs	0.1260
At 500 docs	0.0775
At 1000 docs	0.0494
R—Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2099



Summary Statistics	
Run Number	fsclt3-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	30987
Relevant:	5524
Rel_ret:	1866

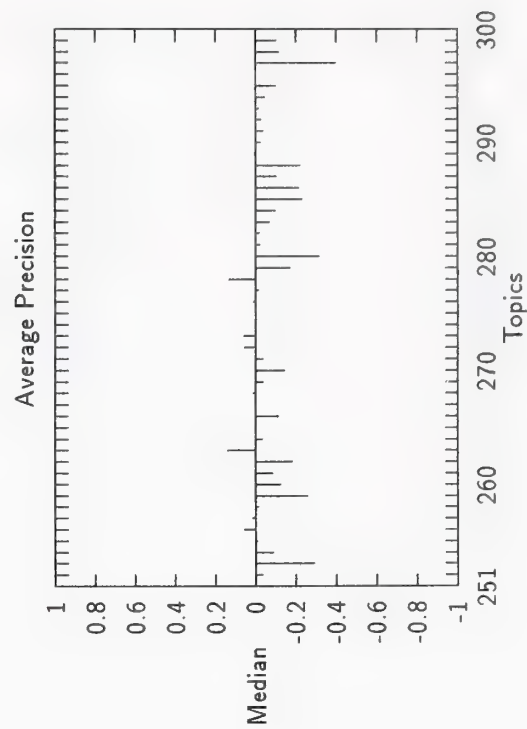
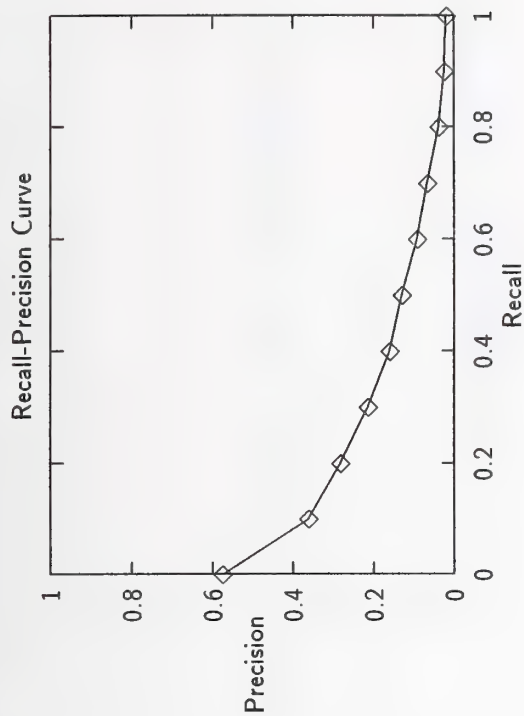
Recall Level Precision Averages	
Recall	Precision
0.00	0.5178
0.10	0.3371
0.20	0.2410
0.30	0.1830
0.40	0.1297
0.50	0.1106
0.60	0.0899
0.70	0.0554
0.80	0.0375
0.90	0.0201
1.00	0.0201
Average precision over all relevant docs	
non-interpolated	0.1368

Document Level Averages	
	Precision
At 5 docs	0.2960
At 10 docs	0.2720
At 15 docs	0.2627
At 20 docs	0.2510
At 30 docs	0.2333
At 100 docs	0.1584
At 200 docs	0.1120
At 500 docs	0.0631
At 1000 docs	0.0373
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1870



Summary Statistics	
Run Number	fsclt4-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2159

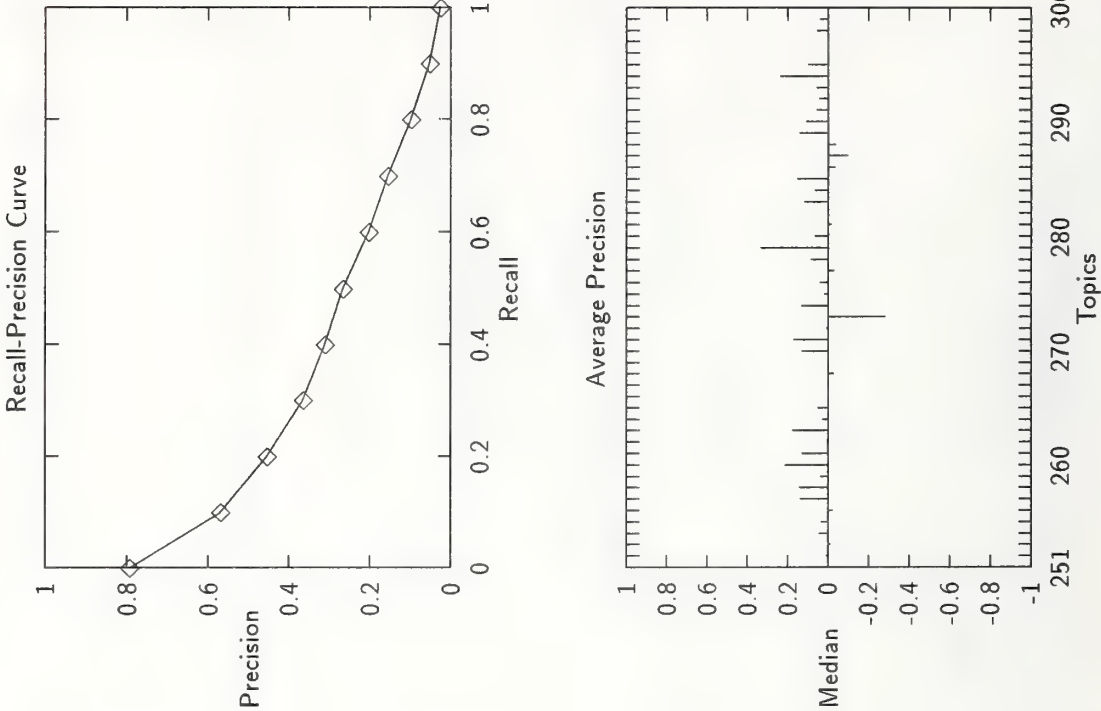
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.5757	At 5 docs	0.3320
0.10	0.3632	At 10 docs	0.3080
0.20	0.2824	At 15 docs	0.2813
0.30	0.2147	At 20 docs	0.2600
0.40	0.1599	At 30 docs	0.2393
0.50	0.1297	At 100 docs	0.1644
0.60	0.0918	At 200 docs	0.1159
0.70	0.0660	At 500 docs	0.0685
0.80	0.0385	At 1000 docs	0.0432
0.90	0.0241	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0187	Exact	0.2006
Average precision over all relevant docs			
non-interpolated	0.1559		



Summary Statistics	
Run Number	genrl3-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	3168

Recall Level Precision Averages	
Recall	Precision
0.00	0.7935
0.10	0.5685
0.20	0.4552
0.30	0.3650
0.40	0.3120
0.50	0.2668
0.60	0.2033
0.70	0.1556
0.80	0.0984
0.90	0.0523
1.00	0.0261
Average precision over all relevant docs	
non-interpolated	0.2784

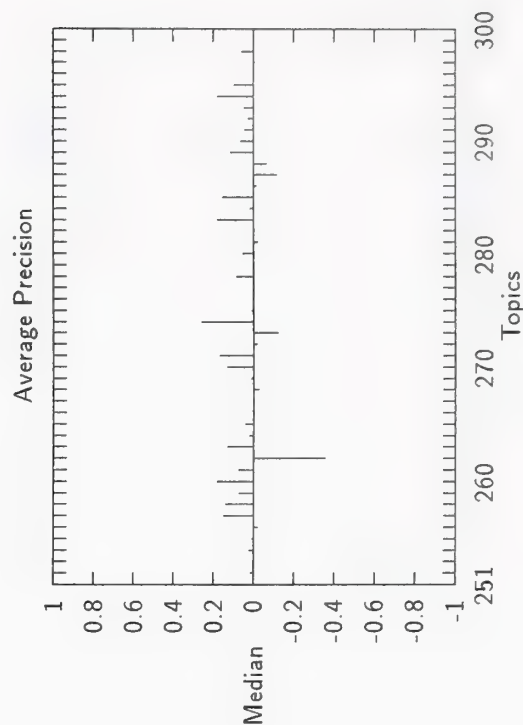
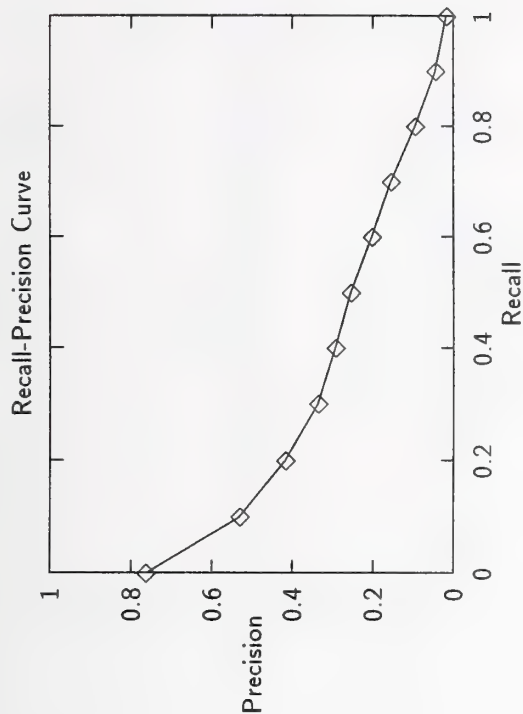
Document Level Averages	
At 5 docs	0.6000
At 10 docs	0.4900
At 15 docs	0.4373
At 20 docs	0.4070
At 30 docs	0.3660
At 100 docs	0.2462
At 200 docs	0.1760
At 500 docs	0.1031
At 1000 docs	0.0634
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3039



Summary Statistics	
Run Number	genrl4-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	3239

Recall Level Precision Averages	
Recall	Precision
0.00	0.7635
0.10	0.5310
0.20	0.4189
0.30	0.3366
0.40	0.2941
0.50	0.2559
0.60	0.2032
0.70	0.1558
0.80	0.0956
0.90	0.0445
1.00	0.0172
Average precision over all relevant docs	
non-interpolated	0.2618

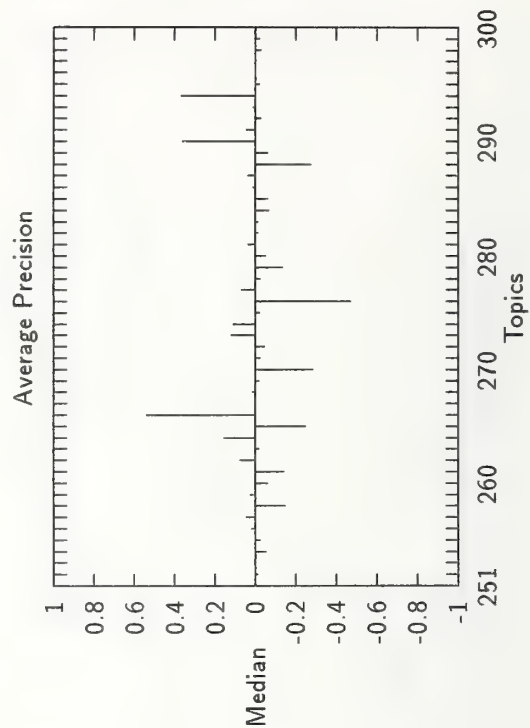
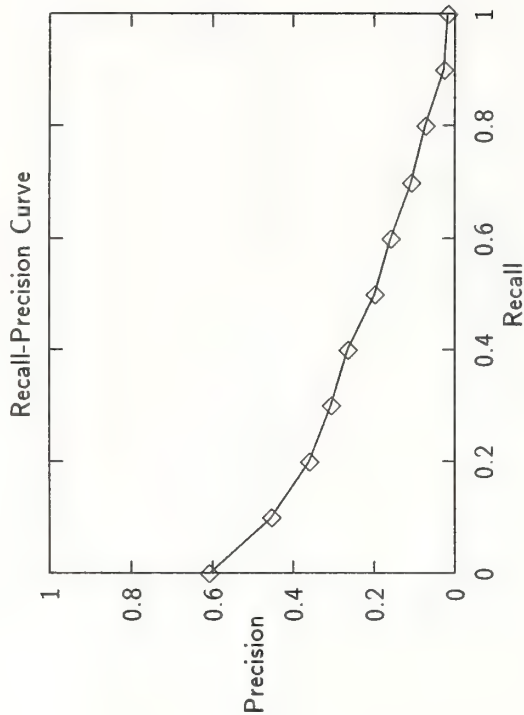
Document Level Averages	
	Precision
At 5 docs	0.5440
At 10 docs	0.4660
At 15 docs	0.4160
At 20 docs	0.3900
At 30 docs	0.3707
At 100 docs	0.2452
At 200 docs	0.1798
At 500 docs	0.1062
At 1000 docs	0.0648
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2860



Summary Statistics	
Run Number	gmu96ma1-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	24178
Relevant:	5524
Rel_ret:	2538

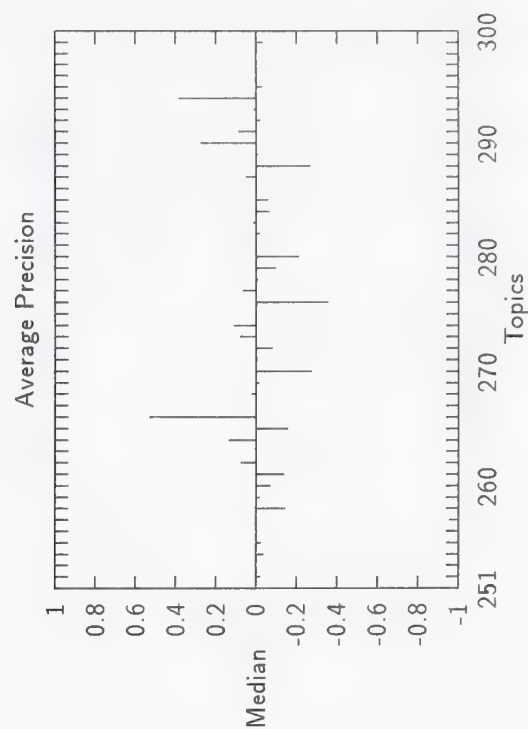
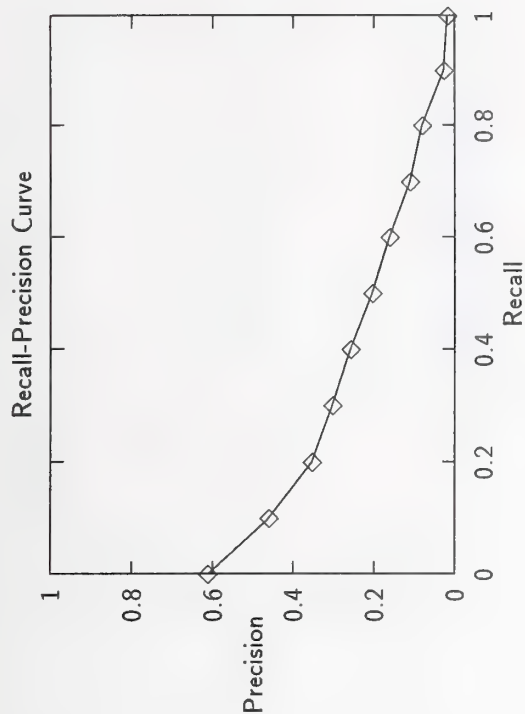
Recall Level Precision Averages	
Recall	Precision
0.00	0.6093
0.10	0.4542
0.20	0.3614
0.30	0.3061
0.40	0.2661
0.50	0.1993
0.60	0.1585
0.70	0.1077
0.80	0.0736
0.90	0.0275
1.00	0.0178
Average precision over all relevant docs	
non-interpolated	0.2147

Document Level Averages	
At 5 docs	0.3840
At 10 docs	0.3800
At 15 docs	0.3733
At 20 docs	0.3650
At 30 docs	0.3547
At 100 docs	0.2454
At 200 docs	0.1710
At 500 docs	0.0913
At 1000 docs	0.0508
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2719



Summary Statistics	
Run Number	gmu96ma2-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	24251
Relevant:	5524
Rel_ret:	2583

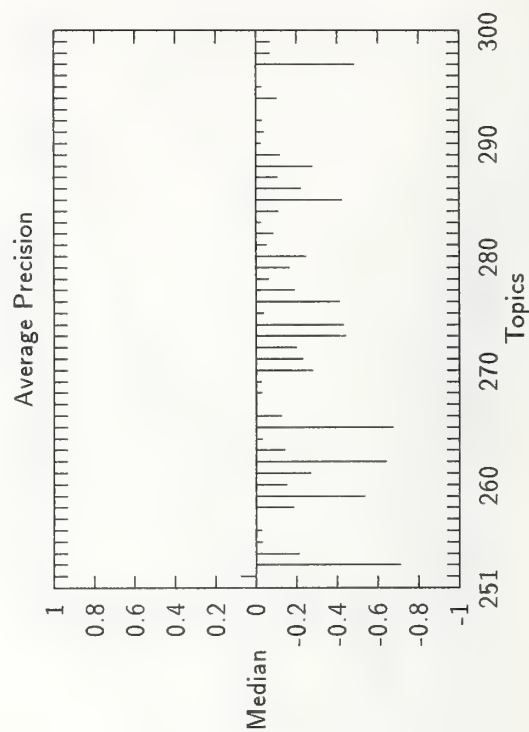
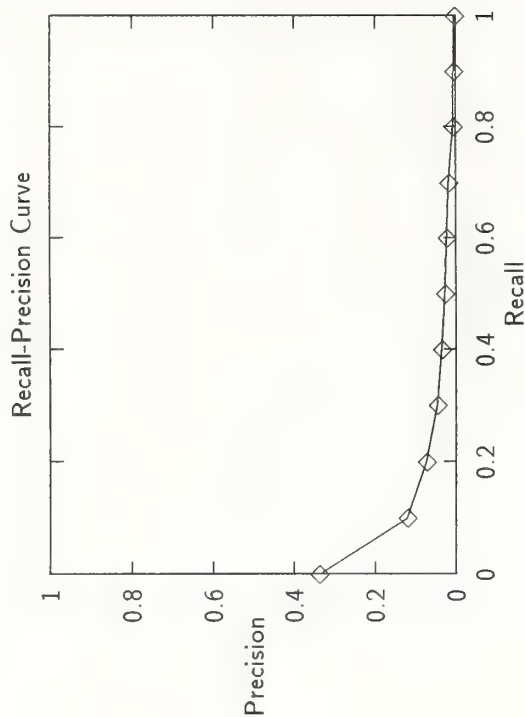
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		
0.00	0.6123	At 5 docs	0.3960
0.10	0.4607	At 10 docs	0.4060
0.20	0.3542	At 15 docs	0.3853
0.30	0.3044	At 20 docs	0.3760
0.40	0.2598	At 30 docs	0.3480
0.50	0.2053	At 100 docs	0.2428
0.60	0.1615	At 200 docs	0.1711
0.70	0.1110	At 500 docs	0.0932
0.80	0.0807	At 1000 docs	0.0517
0.90	0.0268	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0170	Exact	0.2802
Average precision over all relevant docs			
non-interpolated	0.2141		



Summary Statistics	
Run Number	erliA1-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
RelRet:	963

Recall Level Precision Averages	
Recall	Precision
0.00	0.3382
0.10	0.1208
0.20	0.0727
0.30	0.0464
0.40	0.0355
0.50	0.0277
0.60	0.0217
0.70	0.0182
0.80	0.0067
0.90	0.0055
1.00	0.0038
Average precision over all relevant docs	
non-interpolated	0.0449

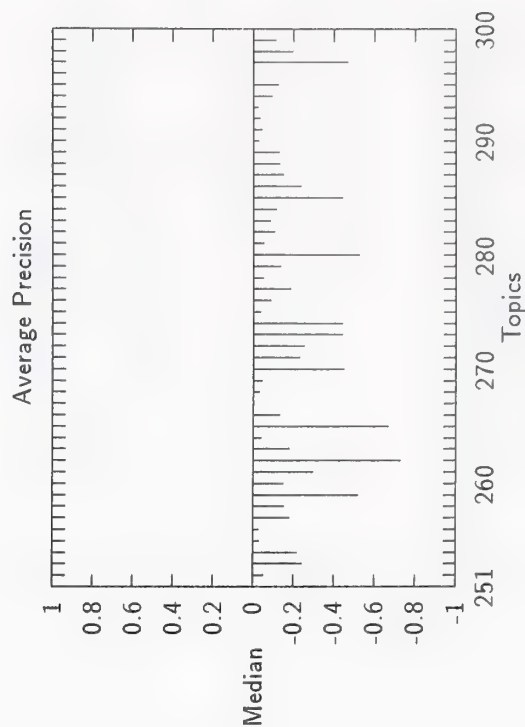
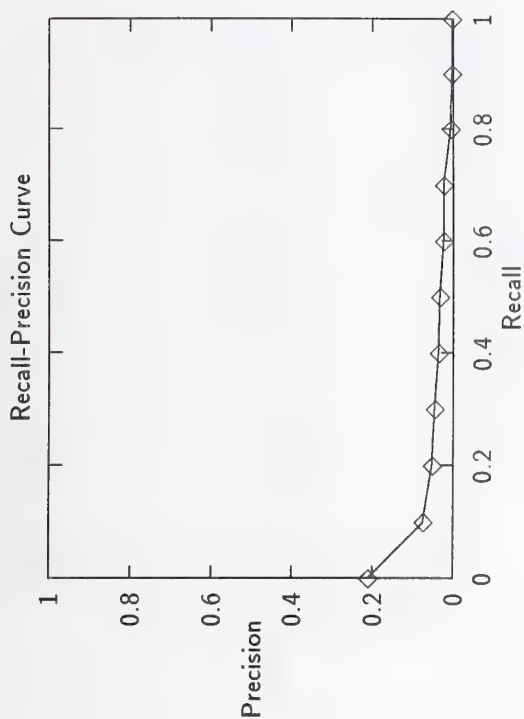
Document Level Averages	
	Precision
At 5 docs	0.1400
At 10 docs	0.1540
At 15 docs	0.1227
At 20 docs	0.1120
At 30 docs	0.0980
At 100 docs	0.0606
At 200 docs	0.0443
At 500 docs	0.0265
At 1000 docs	0.0193
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0677



Summary Statistics	
Run Number	ibmgd2-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel.Ret:	220

Recall Level Precision Averages	
Recall	Precision
0.00	0.2108
0.10	0.0743
0.20	0.0532
0.30	0.0467
0.40	0.0362
0.50	0.0335
0.60	0.0242
0.70	0.0242
0.80	0.0058
0.90	0.0034
1.00	0.0034
Average precision over all relevant docs	
non-interpolated	0.0354

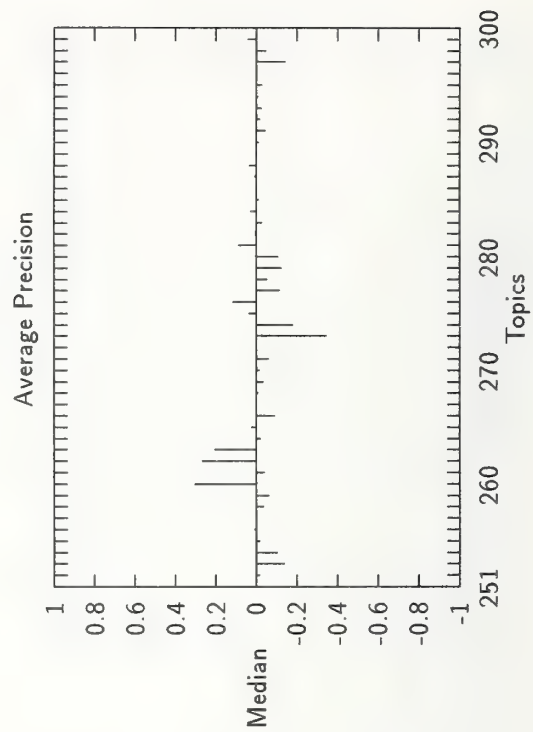
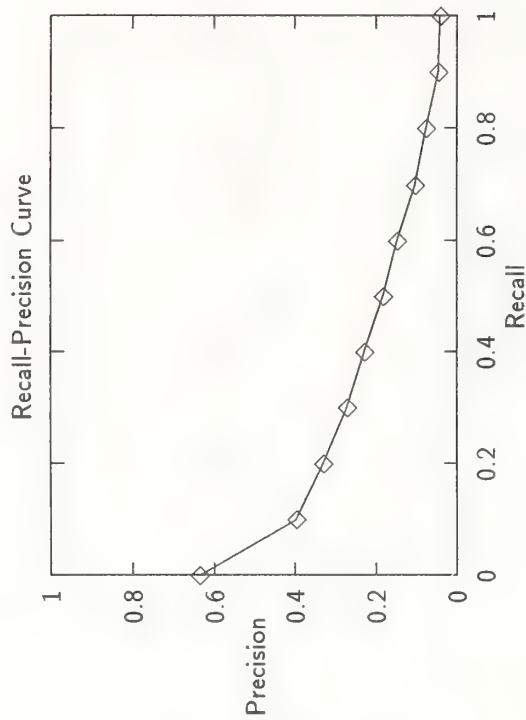
Document Level Averages	
At 5 docs	0.1120
At 10 docs	0.0740
At 15 docs	0.0600
At 20 docs	0.0520
At 30 docs	0.0400
At 100 docs	0.0198
At 200 docs	0.0125
At 500 docs	0.0070
At 1000 docs	0.0044
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0429



Summary Statistics	
Run Number	ibmge2-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2298

Recall Level Precision Averages	
Recall	Precision
0.00	0.6352
0.10	0.3982
0.20	0.3314
0.30	0.2713
0.40	0.2289
0.50	0.1822
0.60	0.1478
0.70	0.1039
0.80	0.0766
0.90	0.0467
1.00	0.0410
Average precision over all relevant docs	
non-interpolated	0.2042

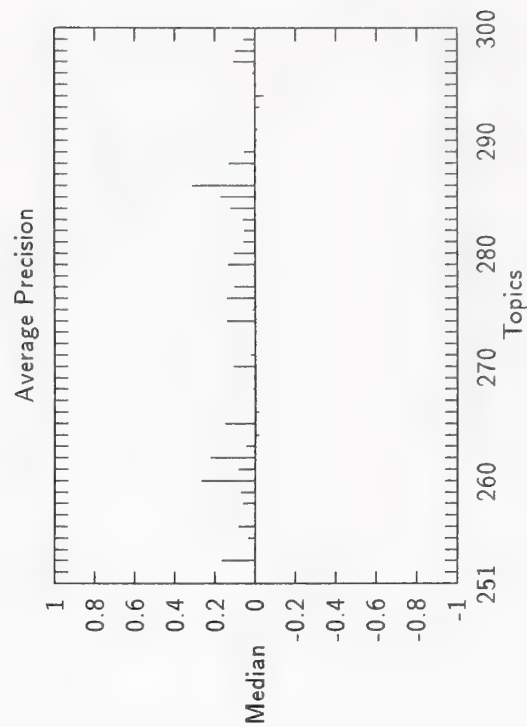
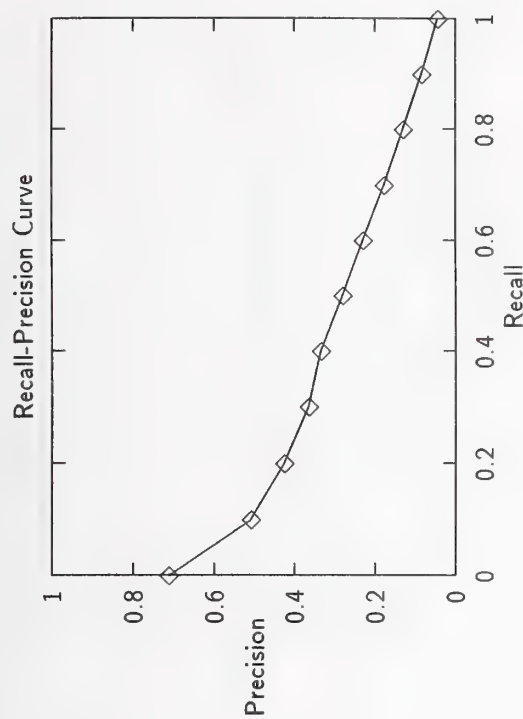
Document Level Averages	
At 5 docs	0.4640
At 10 docs	0.4000
At 15 docs	0.3533
At 20 docs	0.3290
At 30 docs	0.2820
At 100 docs	0.1808
At 200 docs	0.1269
At 500 docs	0.0733
At 1000 docs	0.0460
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2342



Summary Statistics	
Run Number	LNmFull1-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	3087

Recall Level Precision Averages	
Recall	Precision
0.00	0.7126
0.10	0.5082
0.20	0.4259
0.30	0.3655
0.40	0.3347
0.50	0.2816
0.60	0.2299
0.70	0.1780
0.80	0.1305
0.90	0.0841
1.00	0.0446
Average precision over all relevant docs	
non-interpolated	0.2824

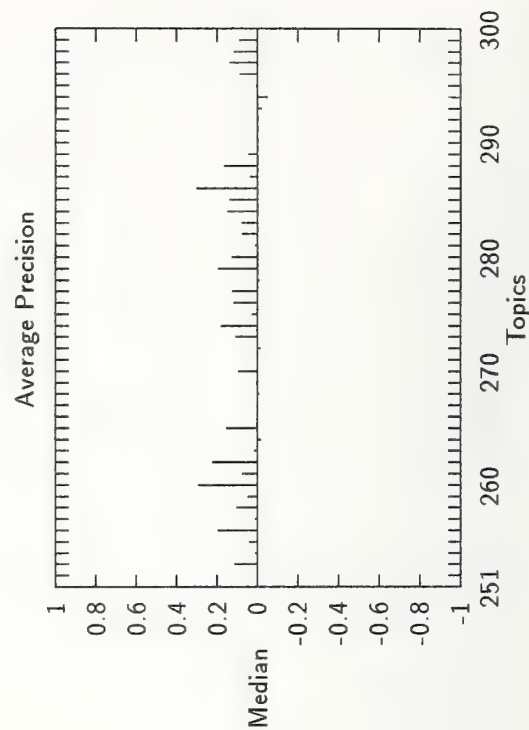
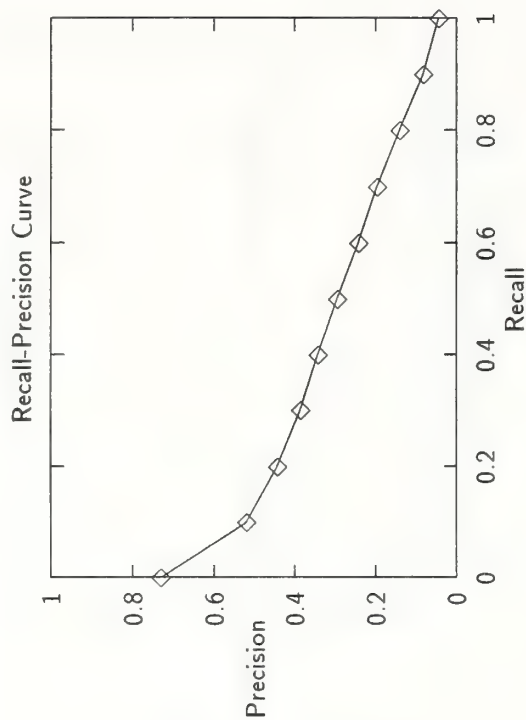
Document Level Averages	
	Precision
At 5 docs	0.5480
At 10 docs	0.4800
At 15 docs	0.4293
At 20 docs	0.3960
At 30 docs	0.3593
At 100 docs	0.2340
At 200 docs	0.1693
At 500 docs	0.0974
At 1000 docs	0.0617
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3129



Summary Statistics	
Run Number	LNmFull2-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel.ret:	3104

Recall Level Precision Averages	
Recall	Precision
0.00	0.7320
0.10	0.5193
0.20	0.4435
0.30	0.3867
0.40	0.3434
0.50	0.2961
0.60	0.2425
0.70	0.1968
0.80	0.1407
0.90	0.0834
1.00	0.0438
Average precision over all relevant docs	
non-interpolated	0.2933

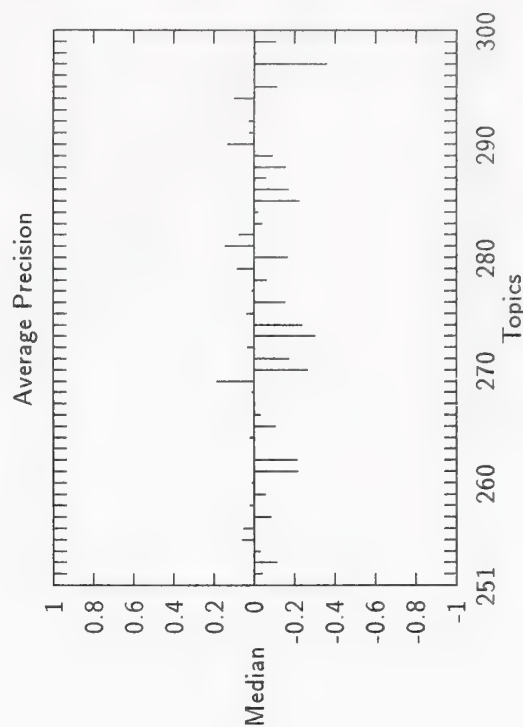
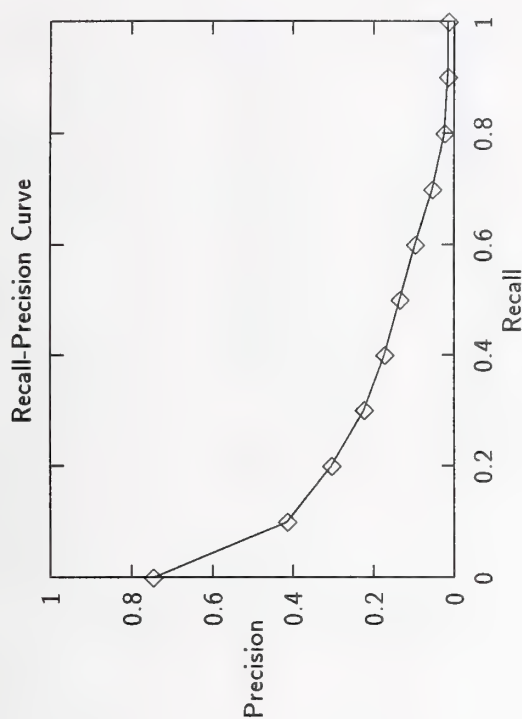
Document Level Averages	
	Precision
At 5 docs	0.5560
At 10 docs	0.4980
At 15 docs	0.4533
At 20 docs	0.4080
At 30 docs	0.3693
At 100 docs	0.2392
At 200 docs	0.1760
At 500 docs	0.1018
At 1000 docs	0.0621
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3231



Summary Statistics	
Run Number	colm1-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	49148
Relevant:	5524
Rel_ret:	2222

Recall Level Precision Averages	
Recall	Precision
0.00	0.7479
0.10	0.4146
0.20	0.3050
0.30	0.2237
0.40	0.1735
0.50	0.1353
0.60	0.0968
0.70	0.0545
0.80	0.0246
0.90	0.0157
1.00	0.0146
Average precision over all relevant docs	
non-interpolated	0.1705

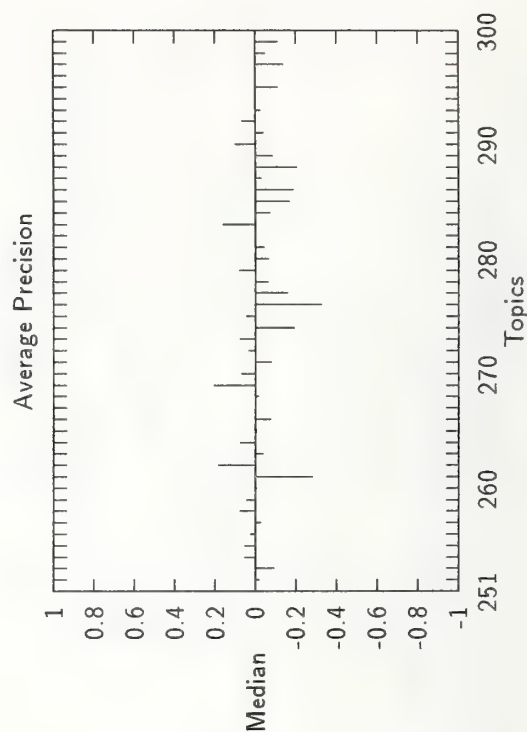
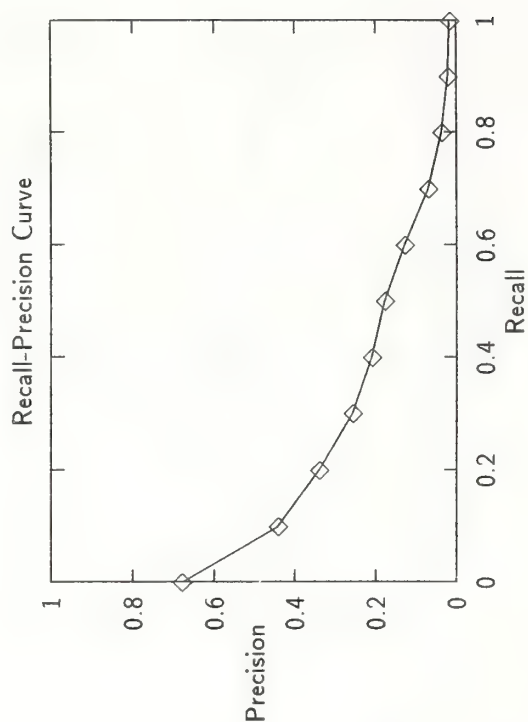
Document Level Averages	
	Precision
At 5 docs	0.4920
At 10 docs	0.4200
At 15 docs	0.3840
At 20 docs	0.3520
At 30 docs	0.3167
At 100 docs	0.1962
At 200 docs	0.1339
At 500 docs	0.0736
At 1000 docs	0.0444
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2351



Summary Statistics	
Run Number	colm4-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2529

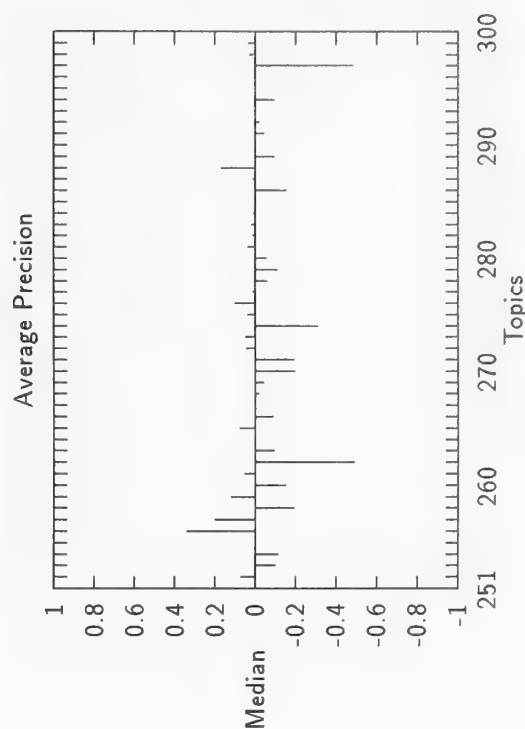
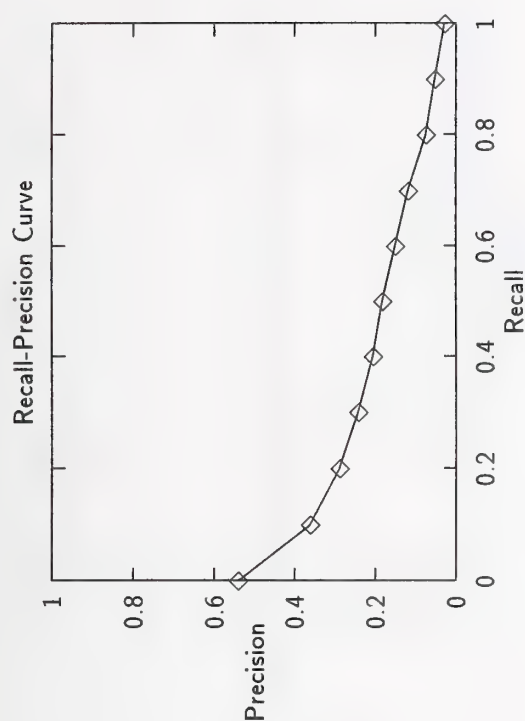
Recall Level Precision Averages	
Recall	Precision
0.00	0.6782
0.10	0.4412
0.20	0.3401
0.30	0.2578
0.40	0.2099
0.50	0.1773
0.60	0.1282
0.70	0.0698
0.80	0.0366
0.90	0.0206
1.00	0.0164
Average precision over all relevant docs	
non-interpolated	0.1951

Document Level Averages	
	Precision
At 5 docs	0.4680
At 10 docs	0.4180
At 15 docs	0.3920
At 20 docs	0.3530
At 30 docs	0.3120
At 100 docs	0.2076
At 200 docs	0.1449
At 500 docs	0.0826
At 1000 docs	0.0506
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2419



Summary Statistics	
Run Number	pircsAM1-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2354

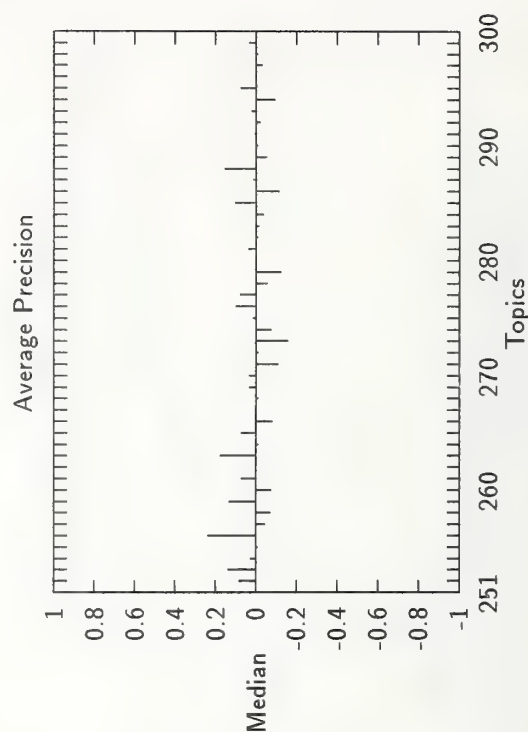
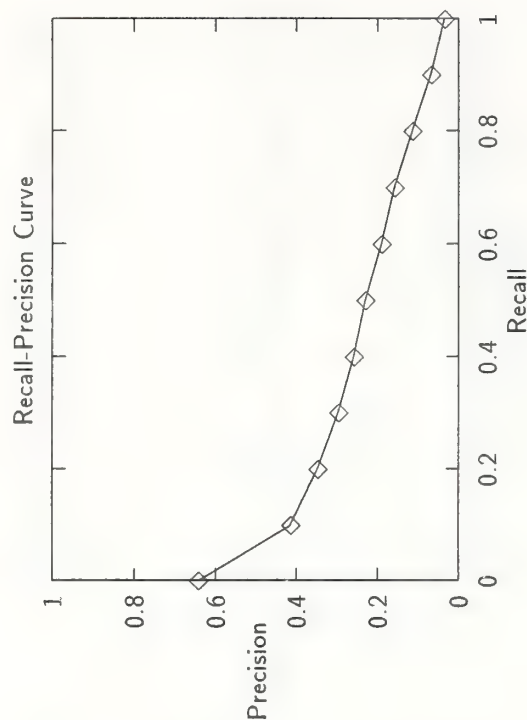
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.5409	At 5 docs	0.3880
0.10	0.3621	At 10 docs	0.3180
0.20	0.2897	At 15 docs	0.3013
0.30	0.2428	At 20 docs	0.2830
0.40	0.2063	At 30 docs	0.2513
0.50	0.1825	At 100 docs	0.1696
0.60	0.1507	At 200 docs	0.1239
0.70	0.1187	At 500 docs	0.0750
0.80	0.0752	At 1000 docs	0.0471
0.90	0.0530	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0283		
Average precision over all relevant docs		Exact	0.2151
non-interpolated	0.1856		



Summary Statistics	
Run Number	pircsAM2-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2801

Recall Level Precision Averages	
Recall	Precision
0.00	0.6430
0.10	0.4146
0.20	0.3481
0.30	0.2979
0.40	0.2585
0.50	0.2303
0.60	0.1906
0.70	0.1591
0.80	0.1140
0.90	0.0685
1.00	0.0347
Average precision over all relevant docs	
non-interpolated	0.2298

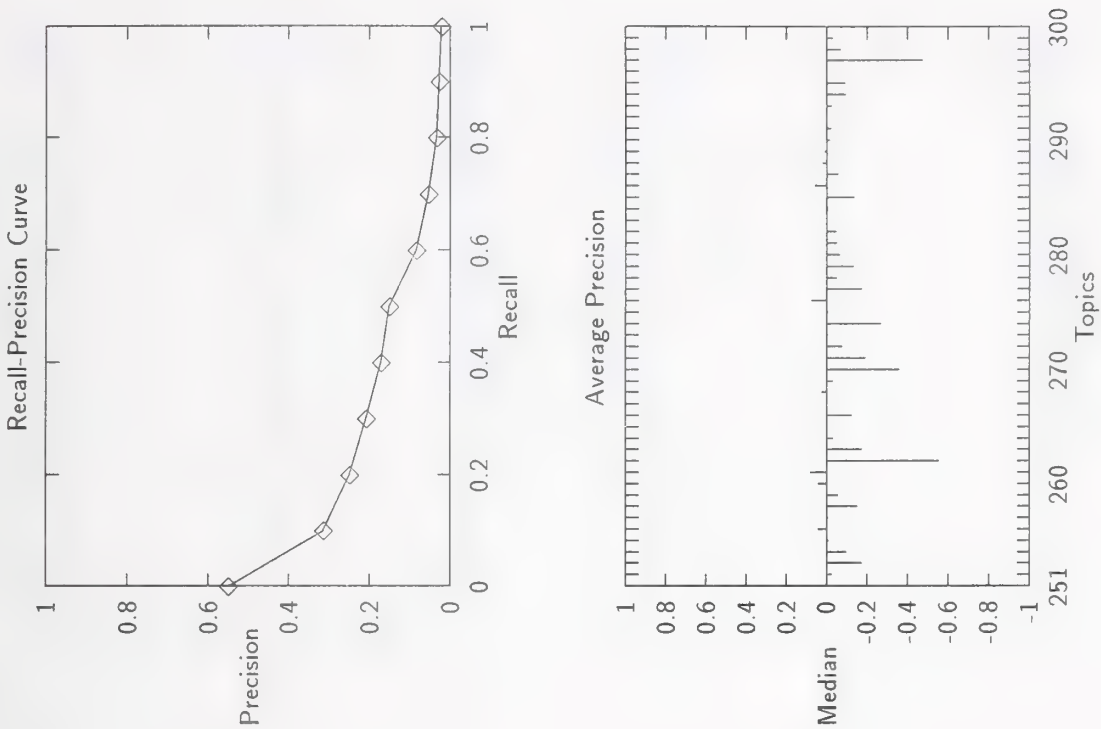
Document Level Averages	
At 5 docs	0.4400
At 10 docs	0.3960
At 15 docs	0.3640
At 20 docs	0.3450
At 30 docs	0.3093
At 100 docs	0.2022
At 200 docs	0.1485
At 500 docs	0.0861
At 1000 docs	0.0560
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2598



Summary Statistics	
Run Number	mds003-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	1867

Recall Level Precision Averages	
Recall	Precision
0.00	0.5496
0.10	0.3146
0.20	0.2493
0.30	0.2079
0.40	0.1713
0.50	0.1517
0.60	0.0839
0.70	0.0543
0.80	0.0330
0.90	0.0269
1.00	0.0208
Average precision over all relevant docs	
non-interpolated	0.1480

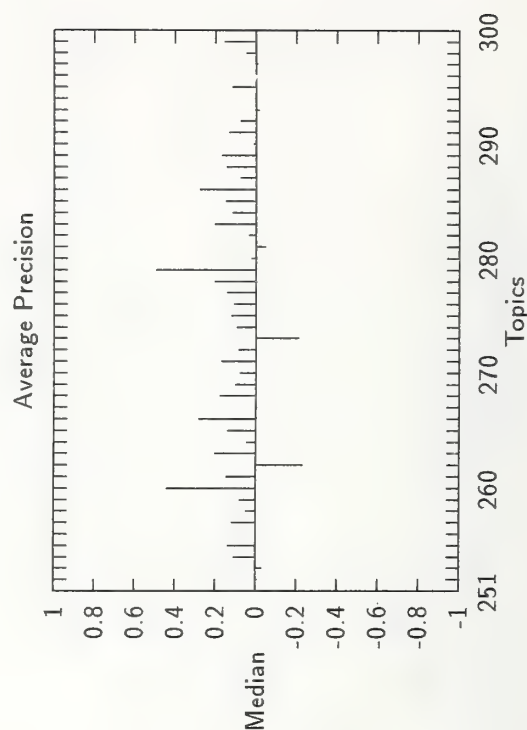
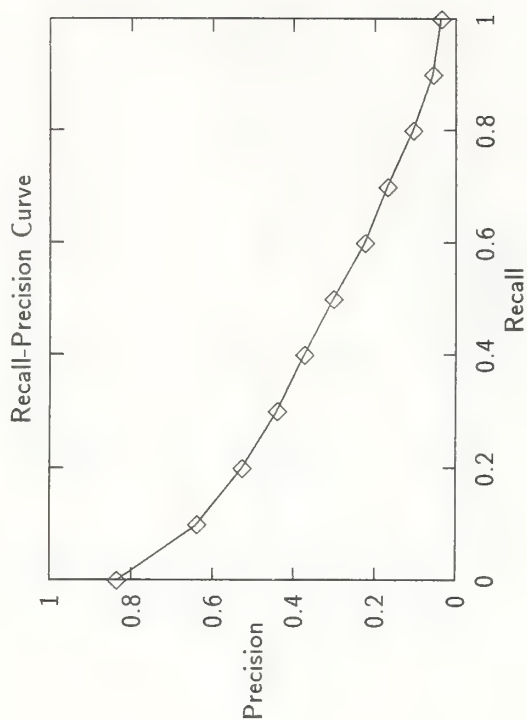
Document Level Averages	
At 5 docs	0.3200
At 10 docs	0.3060
At 15 docs	0.2920
At 20 docs	0.2780
At 30 docs	0.2480
At 100 docs	0.1628
At 200 docs	0.1187
At 500 docs	0.0669
At 1000 docs	0.0373
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1875



Summary Statistics	
Run Number	ETHme1-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	3214

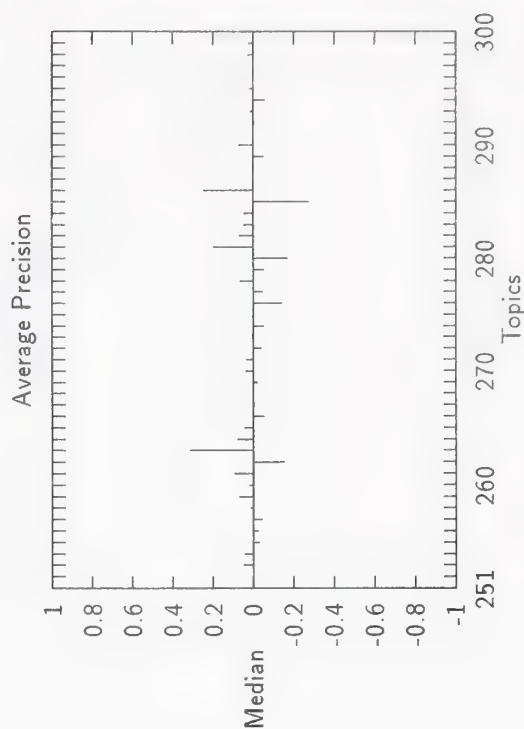
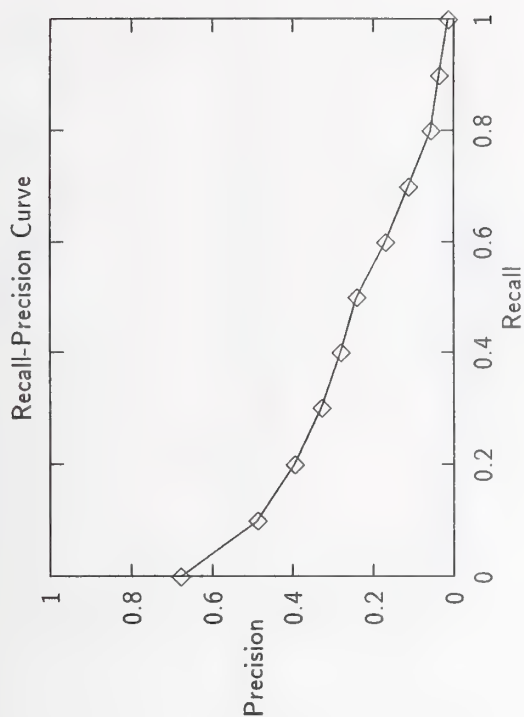
Recall Level Precision Averages	
Recall	Precision
0.00	0.8359
0.10	0.6384
0.20	0.5278
0.30	0.4416
0.40	0.3749
0.50	0.3043
0.60	0.2253
0.70	0.1702
0.80	0.1059
0.90	0.0568
1.00	0.0368
Average precision over all relevant docs	
non-interpolated	0.3165

Document Level Averages	
	Precision
At 5 docs	0.6360
At 10 docs	0.5660
At 15 docs	0.5347
At 20 docs	0.5070
At 30 docs	0.4513
At 100 docs	0.2800
At 200 docs	0.1912
At 500 docs	0.1052
At 1000 docs	0.0643
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3398



Summary Statistics	
Run Number	brkly17-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2945

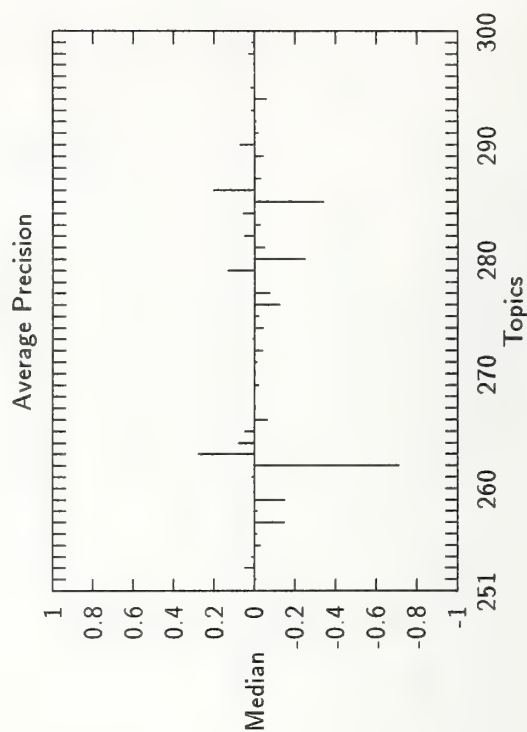
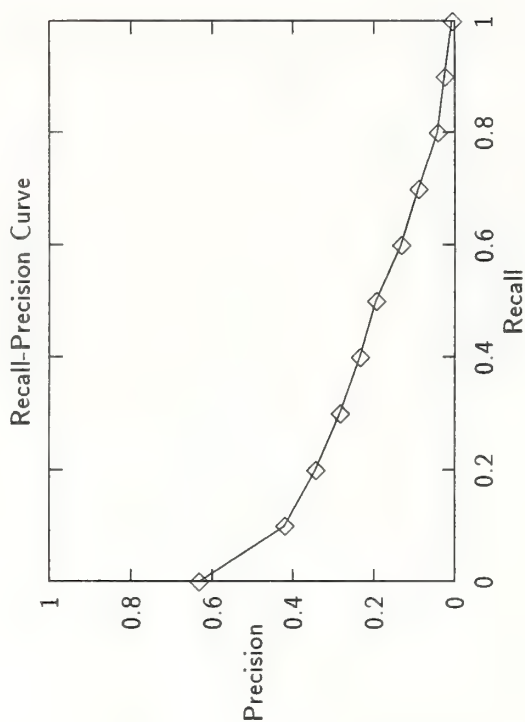
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.6796	At 5 docs	0.5000
0.10	0.4881	At 10 docs	0.4460
0.20	0.3962	At 15 docs	0.4173
0.30	0.3291	At 20 docs	0.3800
0.40	0.2808	At 30 docs	0.3327
0.50	0.2416	At 100 docs	0.2250
0.60	0.1705	At 200 docs	0.1632
0.70	0.1131	At 500 docs	0.0932
0.80	0.0579	At 1000 docs	0.0589
0.90	0.0369	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0146		
Average precision over all relevant docs		Exact	0.2657
non-interpolated			
			0.2346



Summary Statistics	
Run Number	brkly18-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	5524
Rel_ret:	2641

Recall Level Precision Averages	
Recall	Precision
0.00	0.6328
0.10	0.4210
0.20	0.3453
0.30	0.2842
0.40	0.2348
0.50	0.1948
0.60	0.1343
0.70	0.0910
0.80	0.0435
0.90	0.0253
1.00	0.0067
Average precision over all relevant docs	
non-interpolated	0.1983

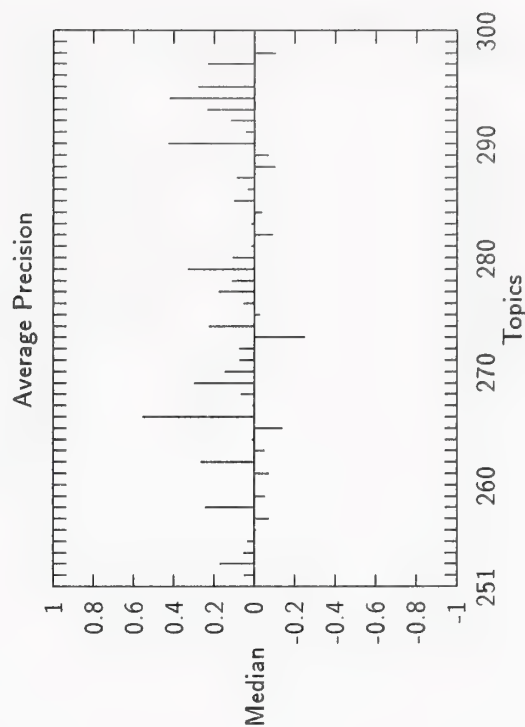
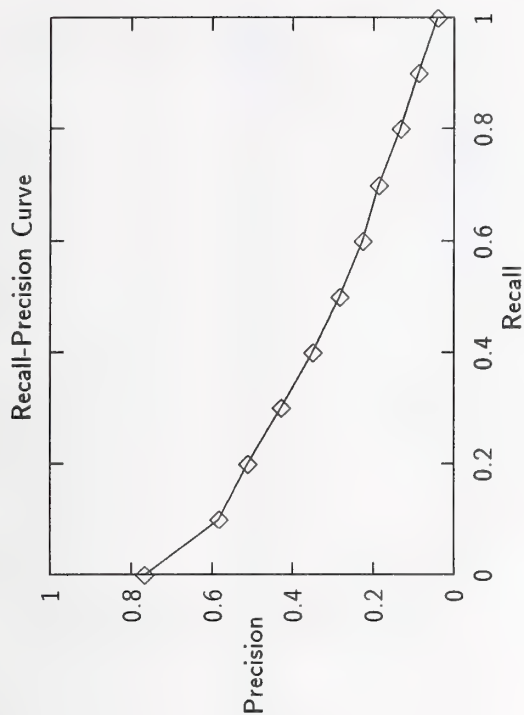
Document Level Averages	
	Precision
At 5 docs	0.4480
At 10 docs	0.3980
At 15 docs	0.3720
At 20 docs	0.3330
At 30 docs	0.3007
At 100 docs	0.2074
At 200 docs	0.1478
At 500 docs	0.0836
At 1000 docs	0.0528
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2489



Summary Statistics	
Run Number	uwgxc0-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	38679
Relevant:	5524
Rel_ret:	3072

Recall Level Precision Averages	
Recall	Precision
0.00	0.7677
0.10	0.5833
0.20	0.5115
0.30	0.4299
0.40	0.3515
0.50	0.2846
0.60	0.2266
0.70	0.1876
0.80	0.1336
0.90	0.0897
1.00	0.0412
Average precision over all relevant docs	
non-interpolated	0.3087

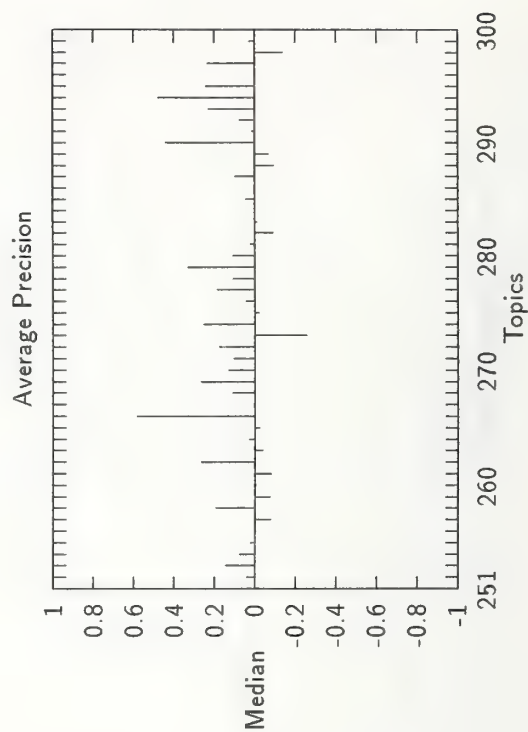
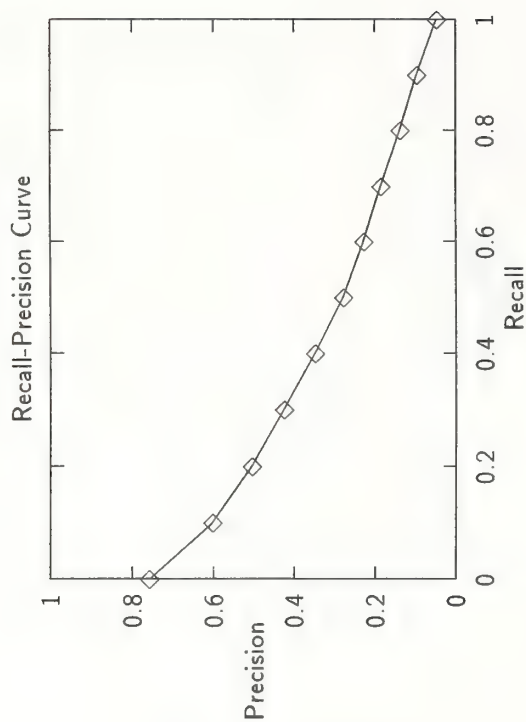
Document Level Averages	
At 5 docs	0.5760
At 10 docs	0.5080
At 15 docs	0.4827
At 20 docs	0.4500
At 30 docs	0.4200
At 100 docs	0.2982
At 200 docs	0.2044
At 500 docs	0.1063
At 1000 docs	0.0614
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3576



Summary Statistics	
Run Number	uwgcx1-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	38679
Relevant:	5524
Rel_ret:	3001

Recall Level Precision Averages	
Recall	Precision
0.00	0.7580
0.10	0.6024
0.20	0.5055
0.30	0.4268
0.40	0.3499
0.50	0.2805
0.60	0.2292
0.70	0.1859
0.80	0.1376
0.90	0.0969
1.00	0.0475
Average precision over all relevant docs	
non-interpolated	0.3098

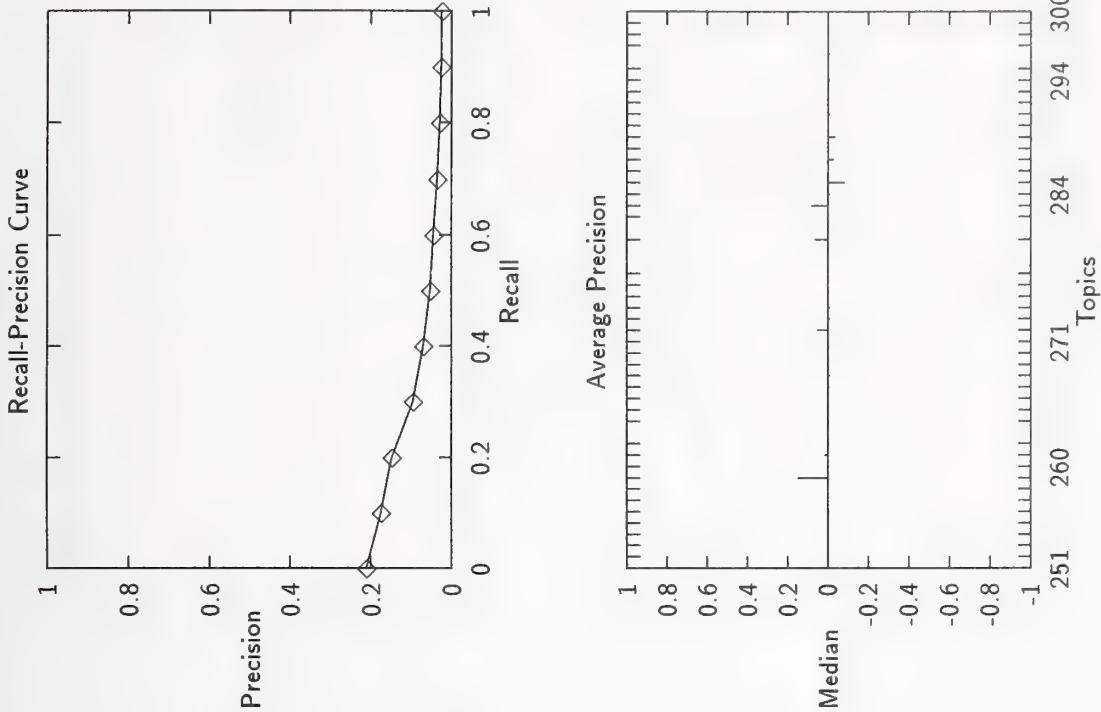
Document Level Averages	
At 5 docs	0.5720
At 10 docs	0.5200
At 15 docs	0.4907
At 20 docs	0.4580
At 30 docs	0.4253
At 100 docs	0.2960
At 200 docs	0.2026
At 500 docs	0.1048
At 1000 docs	0.0600
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3523



Summary Statistics	
Run Number	Ctifr1-category B, automatic, short topic
Number of Topics	45
Total number of documents over all topics	
Retrieved:	44769
Relevant:	1064
Rel.Ret:	378

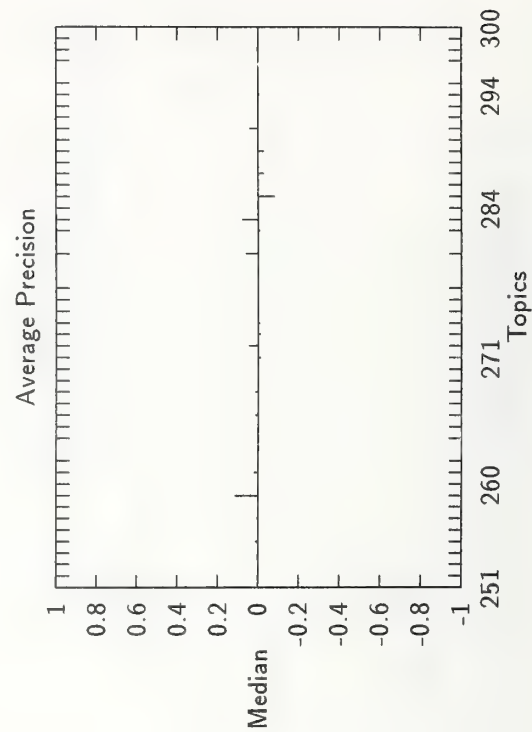
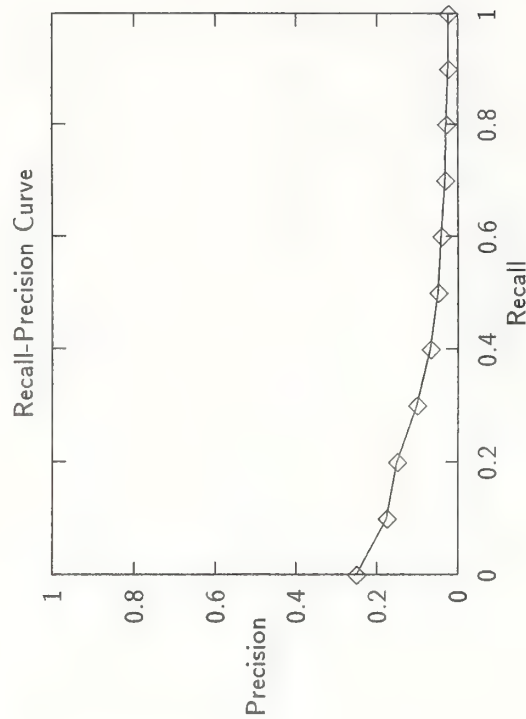
Recall Level Precision Averages	
Recall	Precision
0.00	0.2115
0.10	0.1752
0.20	0.1485
0.30	0.0954
0.40	0.0702
0.50	0.0523
0.60	0.0452
0.70	0.0345
0.80	0.0281
0.90	0.0251
1.00	0.0246
Average precision over all relevant docs	
non-interpolated	0.0743

Document Level Averages	
	Precision
At 5 docs	0.0978
At 10 docs	0.0711
At 15 docs	0.0548
At 20 docs	0.0478
At 30 docs	0.0378
At 100 docs	0.0240
At 200 docs	0.0182
At 500 docs	0.0115
At 1000 docs	0.0084
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0830



Summary Statistics	
Run Number	Ctiff2-category B, automatic, short topic
Number of Topics	45
Total number of documents over all topics	
Retrieved:	44769
Relevant:	1064
RelRet:	403

Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.2505	At 5 docs	0.1022
0.10	0.1760	At 10 docs	0.0711
0.20	0.1505	At 15 docs	0.0607
0.30	0.1019	At 20 docs	0.0522
0.40	0.0686	At 30 docs	0.0474
0.50	0.0493	At 100 docs	0.0271
0.60	0.0416	At 200 docs	0.0198
0.70	0.0321	At 500 docs	0.0125
0.80	0.0280	At 1000 docs	0.0090
0.90	0.0240	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0236		
Average precision over all relevant docs			
non-interpolated	0.0759	Exact	0.0853

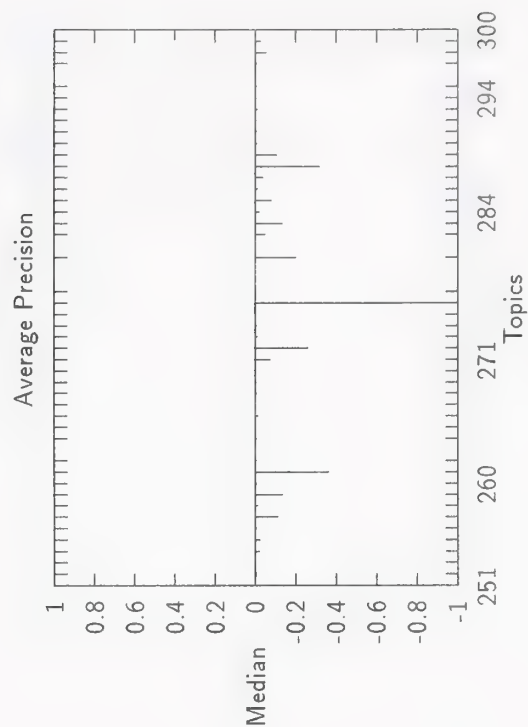
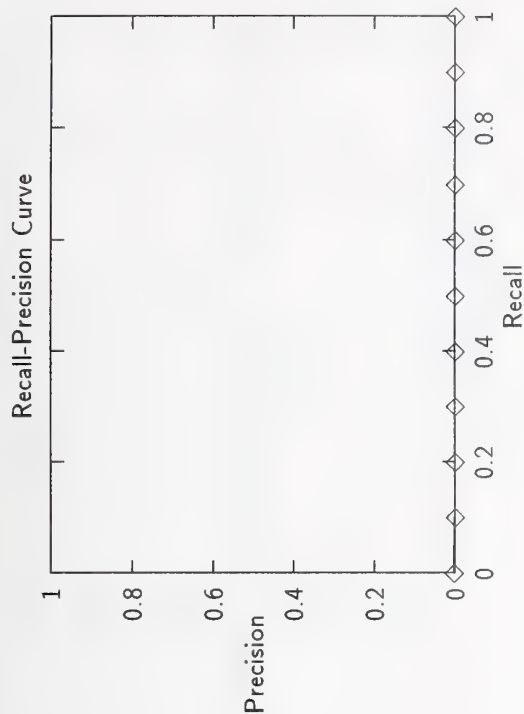


Summary Statistics

Run Number	DCU969-category B, automatic, short topic
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	21

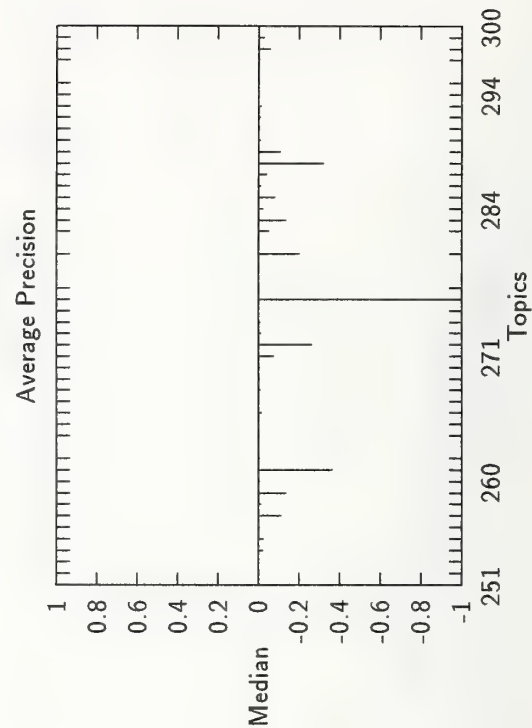
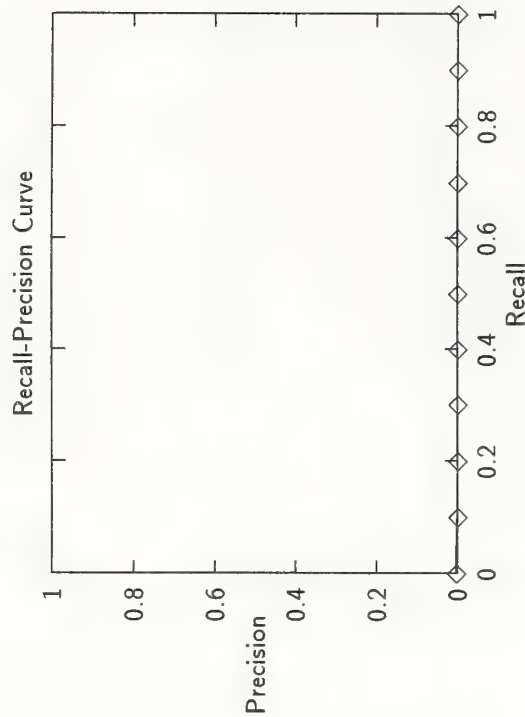
Recall Level Precision Averages	
Recall	Precision
0.00	0.0032
0.10	0.0001
0.20	0.0000
0.30	0.0000
0.40	0.0000
0.50	0.0000
0.60	0.0000
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0001

Document Level Averages	
	Precision
At 5 docs	0.0000
At 10 docs	0.0000
At 15 docs	0.0000
At 20 docs	0.0011
At 30 docs	0.0015
At 100 docs	0.0016
At 200 docs	0.0010
At 500 docs	0.0006
At 1000 docs	0.0005
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0007



Summary Statistics		
Run Number	DCU96B—category B, automatic, short topic	
Number of Topics	45	
Total number of documents over all topics		
Retrieved:	45000	
Relevant:	1064	
Rel_ret:	20	

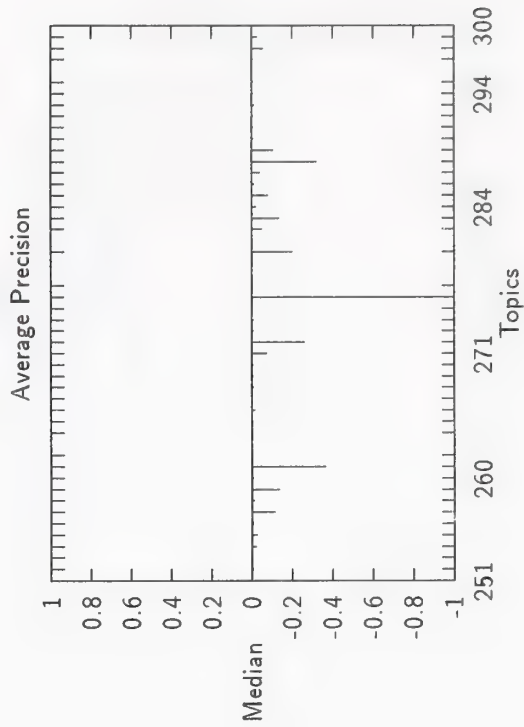
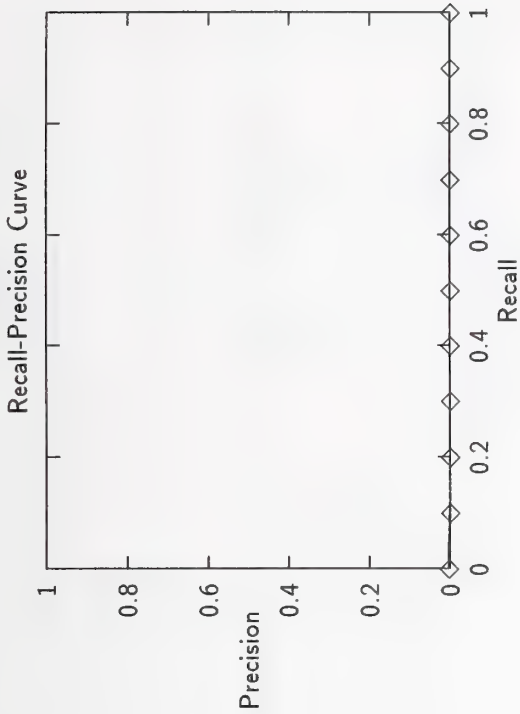
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.0031	At 5 docs	0.0000
0.10	0.0001	At 10 docs	0.0000
0.20	0.0000	At 15 docs	0.0015
0.30	0.0000	At 20 docs	0.0011
0.40	0.0000	At 30 docs	0.0007
0.50	0.0000	At 100 docs	0.0011
0.60	0.0000	At 200 docs	0.0010
0.70	0.0000	At 500 docs	0.0005
0.80	0.0000	At 1000 docs	0.0004
0.90	0.0000	R—Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0000	Exact	0.0008
Average precision over all relevant docs			
non-interpolated		0.0001	



Summary Statistics	
Run Number	DCU96D-category B, automatic, short topic
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	17

Recall Level Precision Averages	
Recall	Precision
0.00	0.0025
0.10	0.0001
0.20	0.0000
0.30	0.0000
0.40	0.0000
0.50	0.0000
0.60	0.0000
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0001

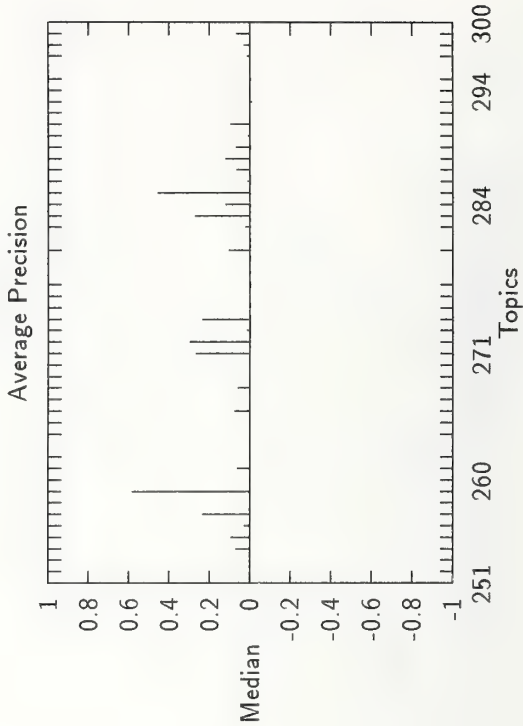
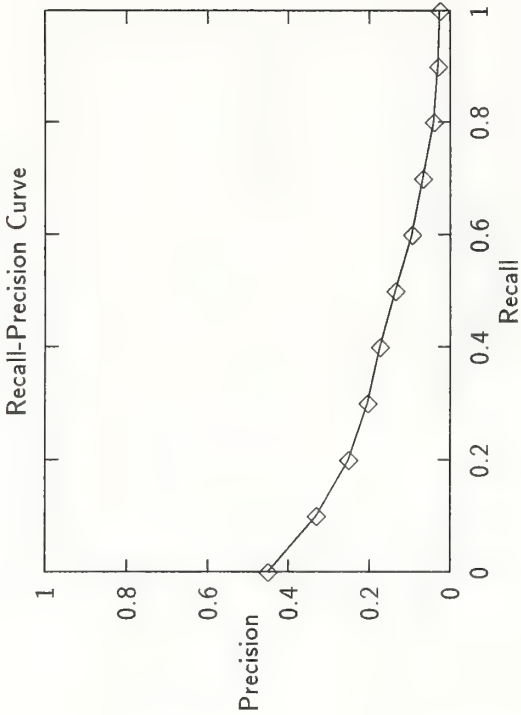
Document Level Averages	
	Precision
At 5 docs	0.0000
At 10 docs	0.0000
At 15 docs	0.0000
At 20 docs	0.0000
At 30 docs	0.0000
At 100 docs	0.0013
At 200 docs	0.0009
At 500 docs	0.0005
At 1000 docs	0.0004
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0007



Summary Statistics		
Run Number	Mercure-as-category B, automatic, short topic	45
Number of Topics		
Total number of documents over all topics		
Retrieved:	45000	
Relevant:	1064	
Rel_ret:	566	

Recall Level Precision Averages	
Recall	Precision
0.00	0.4511
0.10	0.3328
0.20	0.2522
0.30	0.2049
0.40	0.1744
0.50	0.1366
0.60	0.0947
0.70	0.0686
0.80	0.0409
0.90	0.0299
1.00	0.0257
Average precision over all relevant docs	
non-interpolated	0.1499

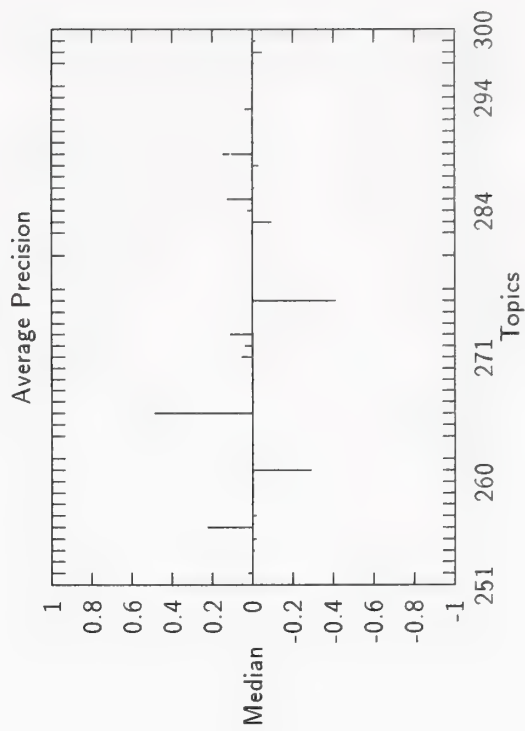
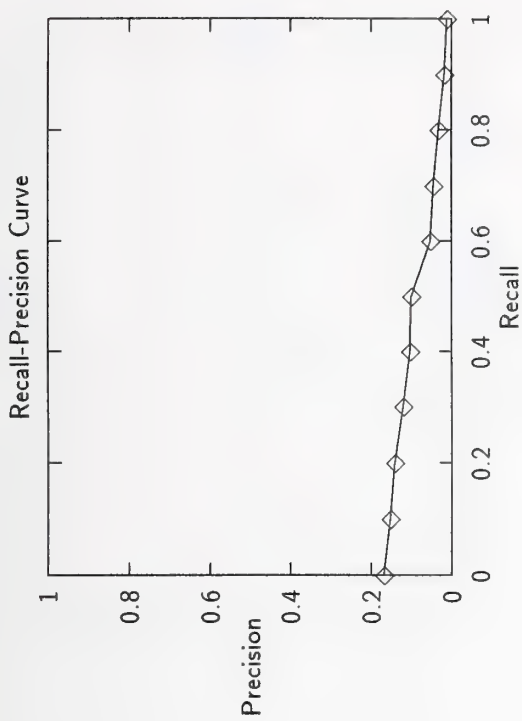
Document Level Averages	
At 5 docs	0.2622
At 10 docs	0.1778
At 15 docs	0.1615
At 20 docs	0.1300
At 30 docs	0.1007
At 100 docs	0.0509
At 200 docs	0.0356
At 500 docs	0.0207
At 1000 docs	0.0126
R—Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1480



Summary Statistics		
Run Number	MONASH—category B, automatic, short topic	
Number of Topics	45	
Total number of documents over all topics		
Retrieved:	45000	
Relevant:	1064	
Rel_ret:	494	

Recall Level Precision Averages	
Recall	Precision
0.00	0.1680
0.10	0.1533
0.20	0.1412
0.30	0.1202
0.40	0.1033
0.50	0.1001
0.60	0.0522
0.70	0.0445
0.80	0.0324
0.90	0.0178
1.00	0.0110
Average precision over all relevant docs	
non-interpolated	0.0779

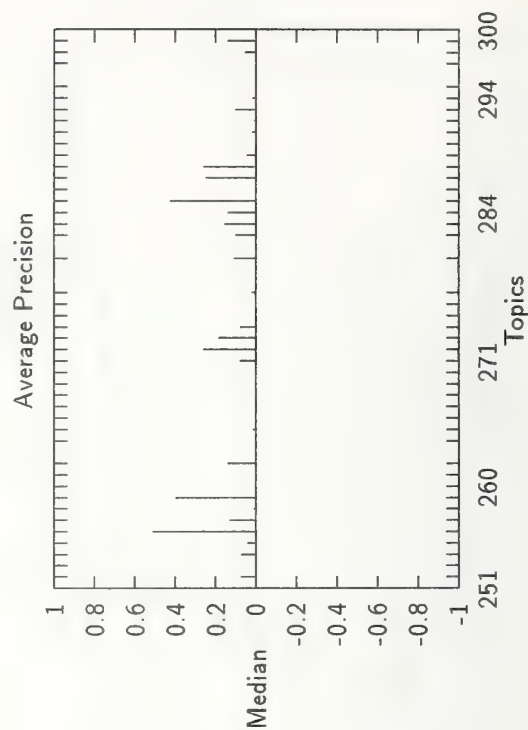
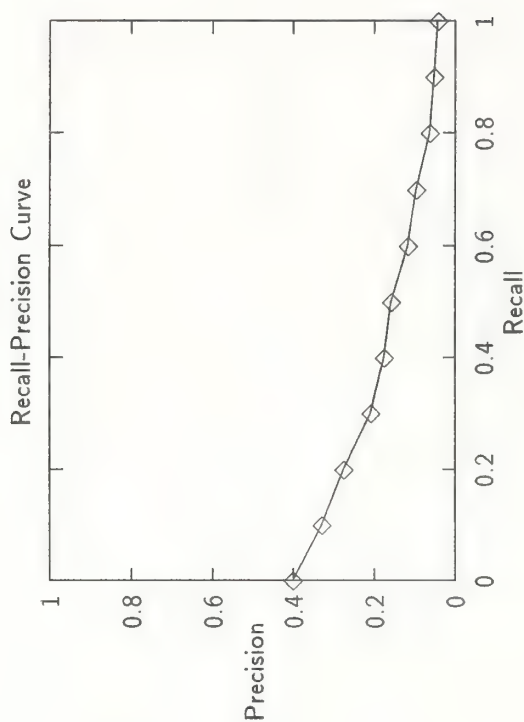
Document Level Averages	
At 5 docs	0.0622
At 10 docs	0.0578
At 15 docs	0.0622
At 20 docs	0.0622
At 30 docs	0.0556
At 100 docs	0.0380
At 200 docs	0.0296
At 500 docs	0.0178
At 1000 docs	0.0110
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0788



Summary Statistics	
Run Number	UniNE7-category B, automatic, short topic
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	643

Recall Level Precision Averages	
Recall	Precision
0.00	0.4027
0.10	0.3306
0.20	0.2768
0.30	0.2096
0.40	0.1760
0.50	0.1587
0.60	0.1182
0.70	0.0968
0.80	0.0628
0.90	0.0523
1.00	0.0422
Average precision over all relevant docs	
non-interpolated	0.1597

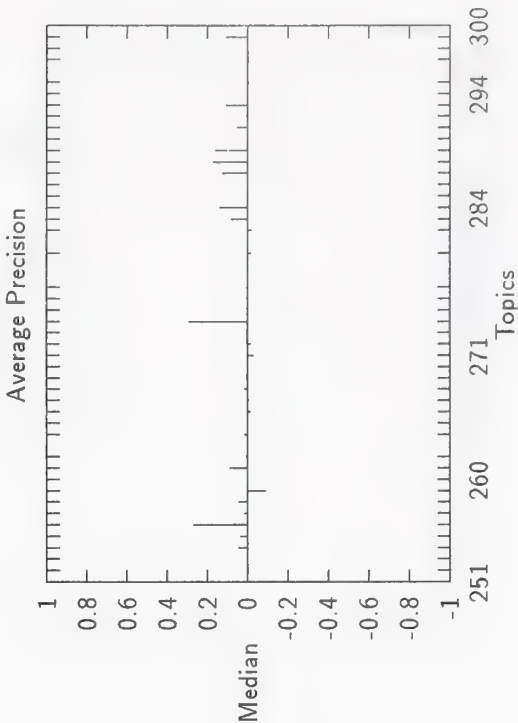
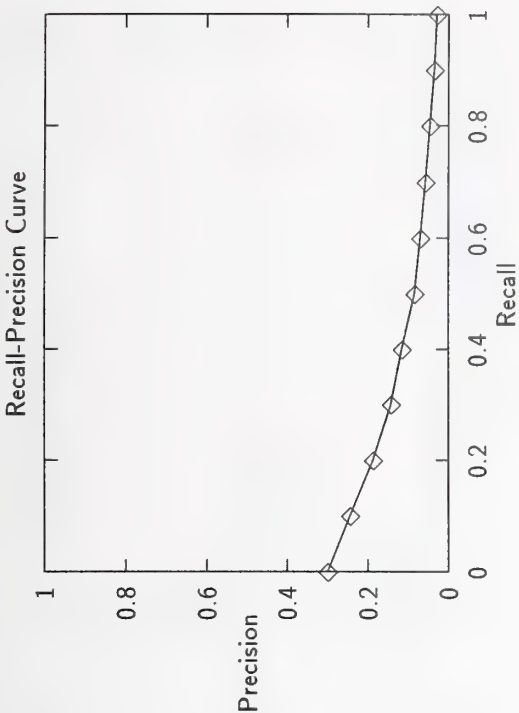
Document Level Averages	
	Precision
At 5 docs	0.2222
At 10 docs	0.1778
At 15 docs	0.1481
At 20 docs	0.1322
At 30 docs	0.1067
At 100 docs	0.0604
At 200 docs	0.0406
At 500 docs	0.0233
At 1000 docs	0.0143
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1722



Summary Statistics	
Run Number	sdmix2-category B, automatic, short topic
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel.ret:	577

Recall Level Precision Averages	
Recall	Precision
0.00	0.3003
0.10	0.2451
0.20	0.1882
0.30	0.1436
0.40	0.1157
0.50	0.0849
0.60	0.0695
0.70	0.0573
0.80	0.0465
0.90	0.0351
1.00	0.0292
Average precision over all relevant docs	
non-interpolated	0.1090

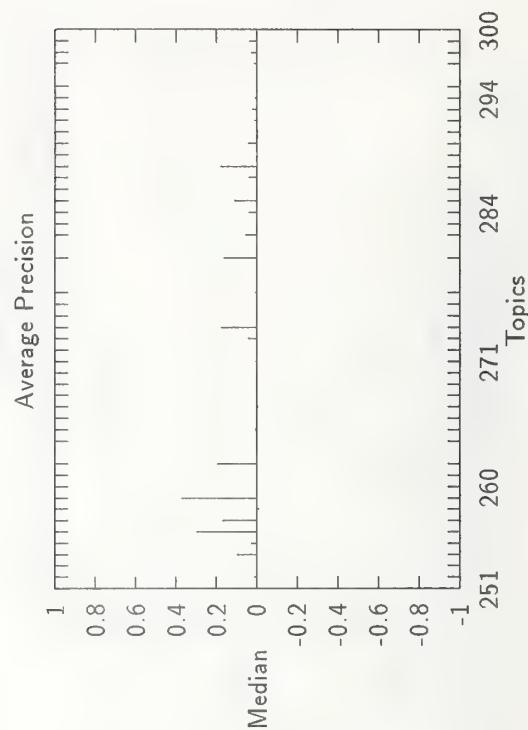
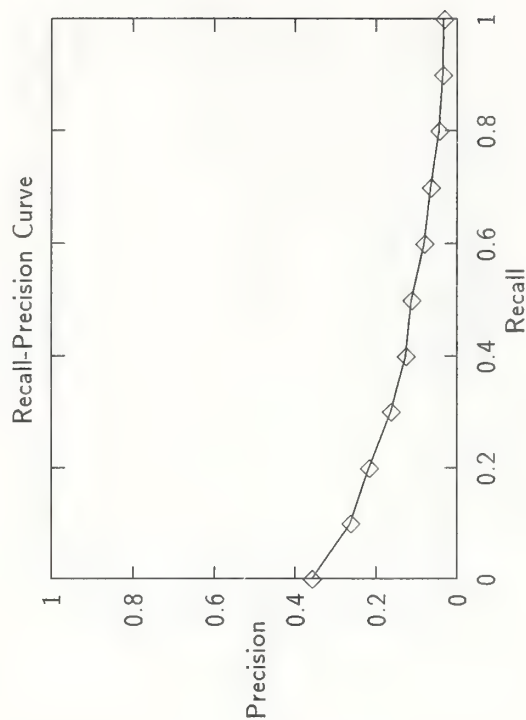
Document Level Averages	
	Precision
At 5 docs	0.1556
At 10 docs	0.1289
At 15 docs	0.1126
At 20 docs	0.1011
At 30 docs	0.0837
At 100 docs	0.0511
At 200 docs	0.0350
At 500 docs	0.0205
At 1000 docs	0.0128
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1333



Summary Statistics	
Run Number	glair4-category B, automatic, short topic
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	494

Recall Level Precision Averages	
Recall	Precision
0.00	0.3595
0.10	0.2647
0.20	0.2175
0.30	0.1645
0.40	0.1267
0.50	0.1124
0.60	0.0807
0.70	0.0658
0.80	0.0446
0.90	0.0349
1.00	0.0318
Average precision over all relevant docs	
non-interpolated	0.1221

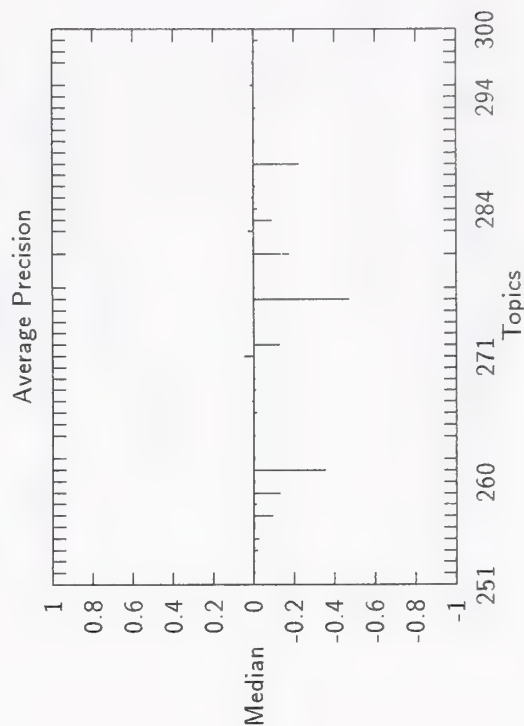
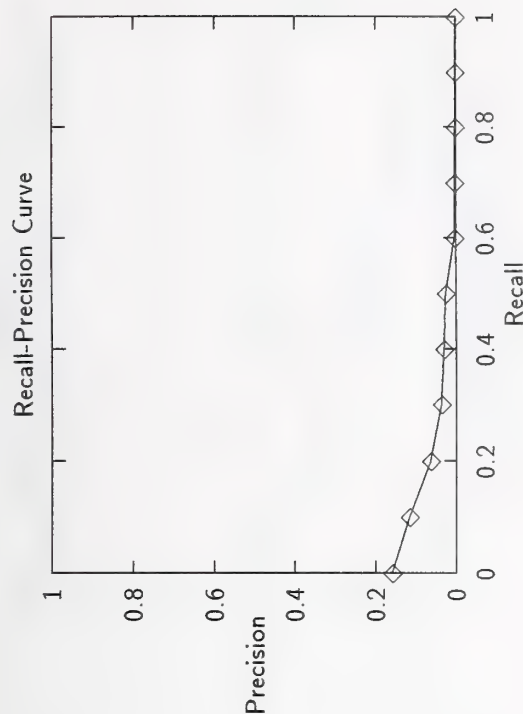
Document Level Averages	
	Precision
At 5 docs	0.1911
At 10 docs	0.1444
At 15 docs	0.1215
At 20 docs	0.1044
At 30 docs	0.0948
At 100 docs	0.0493
At 200 docs	0.0322
At 500 docs	0.0180
At 1000 docs	0.0110
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1418



Summary Statistics	
Run Number	umcpa1-category B, automatic, short topic
Number of Topics	45
Total number of documents over all topics	
Retrieved:	40317
Relevant:	1064
RelRet:	213

Recall Level Precision Averages	
Recall	Precision
0.00	0.1589
0.10	0.1154
0.20	0.0628
0.30	0.0368
0.40	0.0299
0.50	0.0249
0.60	0.0028
0.70	0.0028
0.80	0.0022
0.90	0.0022
1.00	0.0022
Average precision over all relevant docs	
non-interpolated	0.0322

Document Level Averages	
At 5 docs	0.0622
At 10 docs	0.0556
At 15 docs	0.0474
At 20 docs	0.0444
At 30 docs	0.0348
At 100 docs	0.0204
At 200 docs	0.0130
At 500 docs	0.0076
At 1000 docs	0.0047
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0482

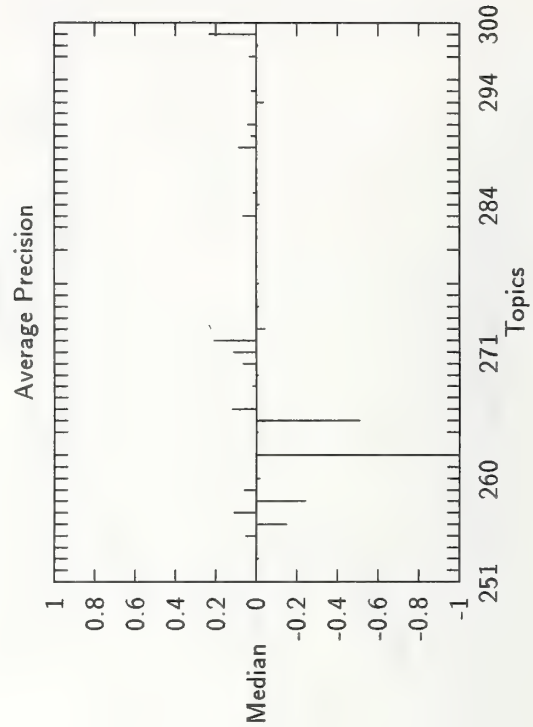
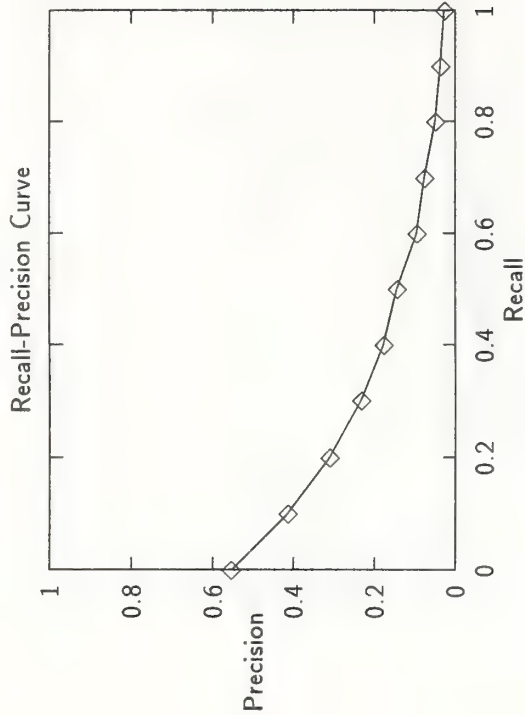


Summary Statistics	
Run Number	Mercuré-al-category B, automatic, long topic
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	630

Recall Level Precision Averages	
Recall	Precision
0.00	0.5541
0.10	0.4148
0.20	0.3121
0.30	0.2340
0.40	0.1781
0.50	0.1448
0.60	0.0949
0.70	0.0770
0.80	0.0493
0.90	0.0359
1.00	0.0268

Average precision over all relevant docs	
non-interpolated	0.1728

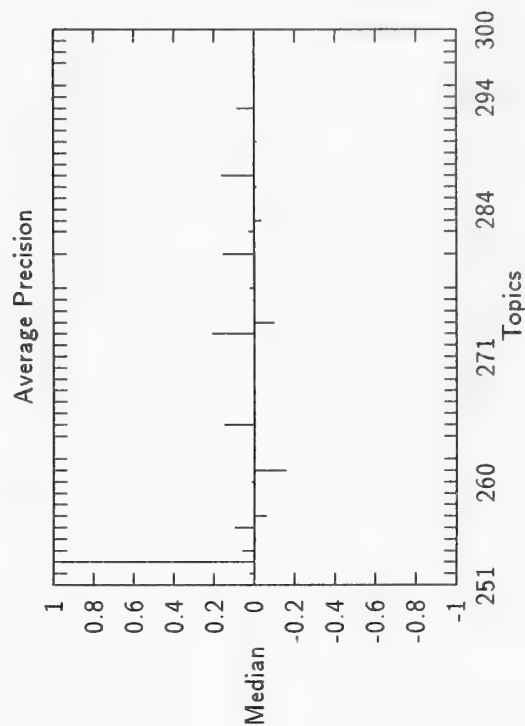
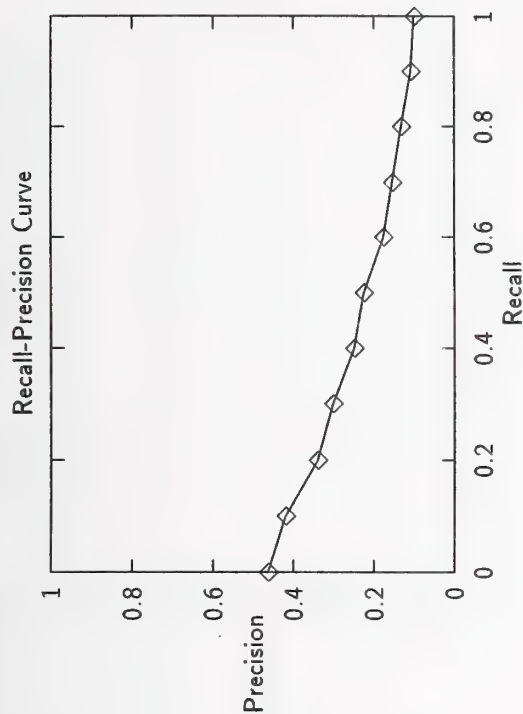
Document Level Averages	
	Precision
At 5 docs	0.2756
At 10 docs	0.2200
At 15 docs	0.1748
At 20 docs	0.1589
At 30 docs	0.1230
At 100 docs	0.0633
At 200 docs	0.0416
At 500 docs	0.0228
At 1000 docs	0.0140
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1784



Summary Statistics	
Run Number	UniNE8-category B, automatic, long topic
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	710

Recall Level Precision Averages	
Recall	Precision
0.00	0.4623
0.10	0.4203
0.20	0.3408
0.30	0.3018
0.40	0.2502
0.50	0.2255
0.60	0.1771
0.70	0.1548
0.80	0.1314
0.90	0.1089
1.00	0.0994
Average precision over all relevant docs	
non-interpolated	0.2272

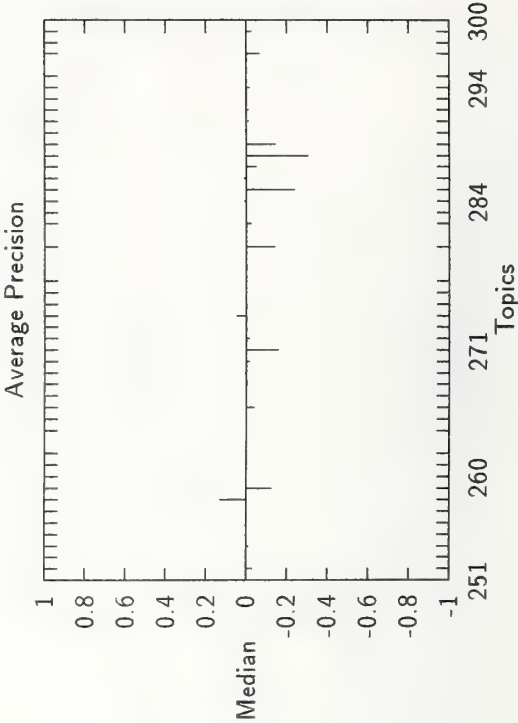
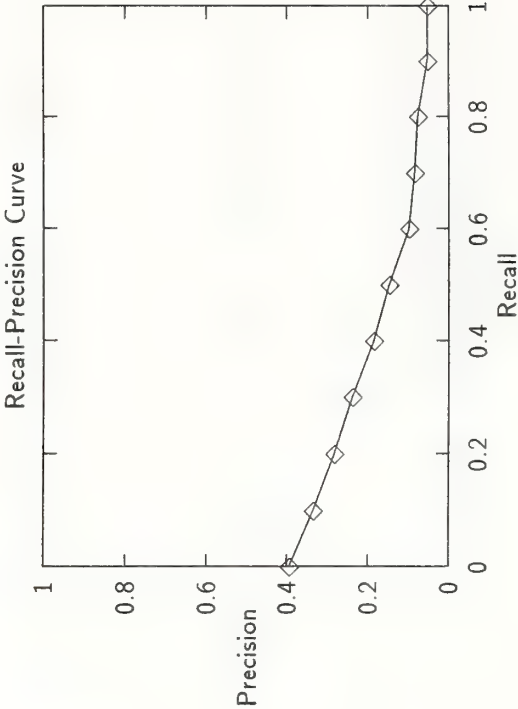
Document Level Averages	
	Precision
At 5 docs	0.2533
At 10 docs	0.2156
At 15 docs	0.1822
At 20 docs	0.1578
At 30 docs	0.1296
At 100 docs	0.0716
At 200 docs	0.0483
At 500 docs	0.0264
At 1000 docs	0.0158
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2462



Summary Statistics	
Run Number	sdmix1-category B, automatic, long topic
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	576

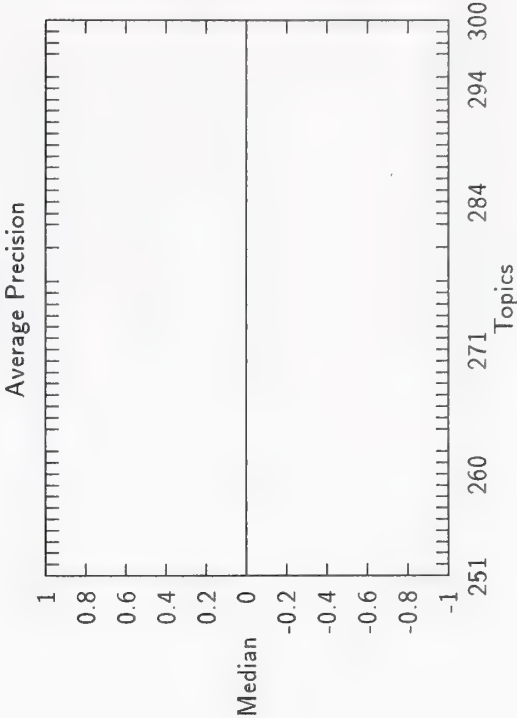
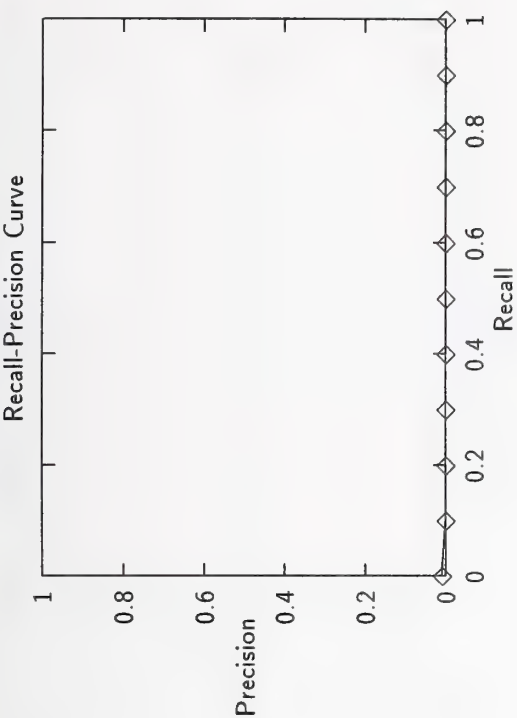
Recall Level Precision Averages	
Recall	Precision
0.00	0.3942
0.10	0.3353
0.20	0.2835
0.30	0.2377
0.40	0.1854
0.50	0.1471
0.60	0.0979
0.70	0.0843
0.80	0.0769
0.90	0.0530
1.00	0.0517
Average precision over all relevant docs	
non-interpolated	0.1633

Document Level Averages	
	Precision
At 5 docs	0.2044
At 10 docs	0.1711
At 15 docs	0.1556
At 20 docs	0.1344
At 30 docs	0.1030
At 100 docs	0.0544
At 200 docs	0.0364
At 500 docs	0.0203
At 1000 docs	0.0128
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1877



Summary Statistics	
Run Number	DCU968-category B, manual
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	21

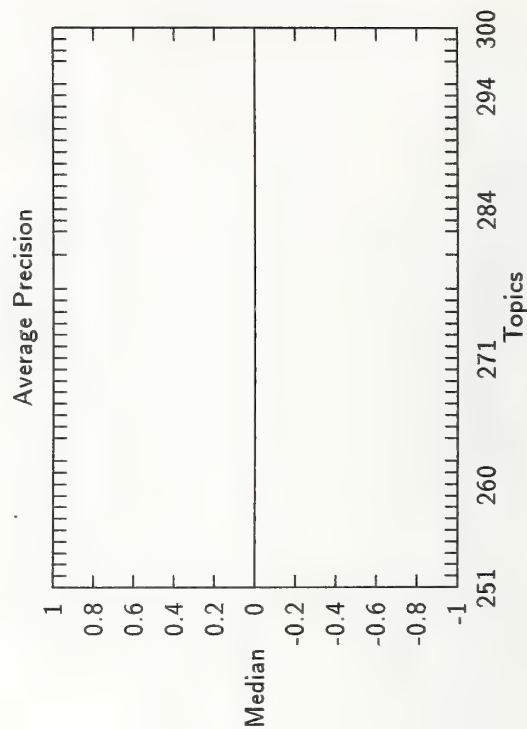
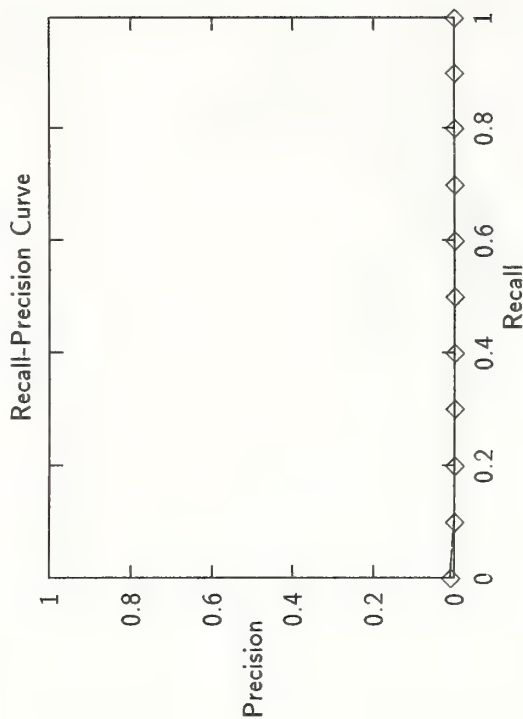
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.0093	At 5 docs	0.0044
0.10	0.0000	At 10 docs	0.0022
0.20	0.0000	At 15 docs	0.0015
0.30	0.0000	At 20 docs	0.0011
0.40	0.0000	At 30 docs	0.0007
0.50	0.0000	At 100 docs	0.0009
0.60	0.0000	At 200 docs	0.0011
0.70	0.0000	At 500 docs	0.0006
0.80	0.0000	At 1000 docs	0.0005
0.90	0.0000	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0000	Exact	0.0011
Average precision over all relevant docs			
non-interpolated		0.0001	



Summary Statistics	
Run Number	DCU96A-category B, manual
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	21

Recall Level Precision Averages	
Recall	Precision
0.00	0.0089
0.10	0.0001
0.20	0.0000
0.30	0.0000
0.40	0.0000
0.50	0.0000
0.60	0.0000
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0001

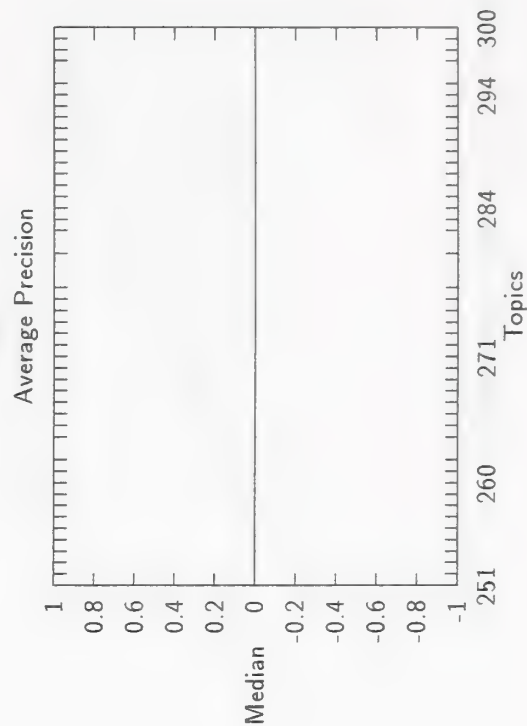
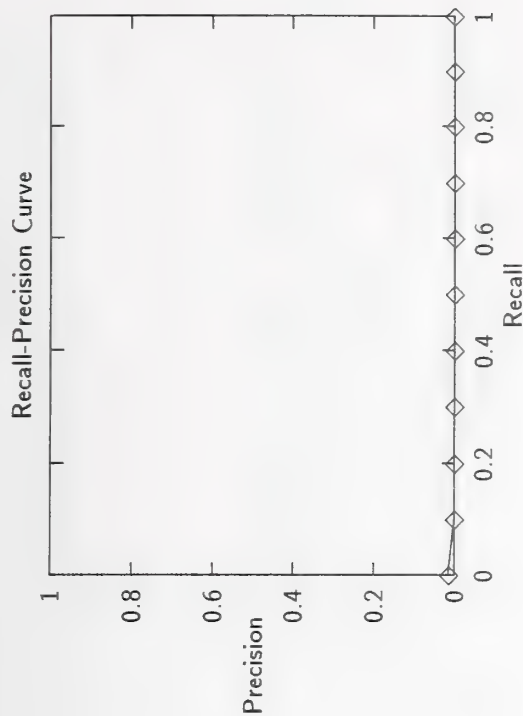
Document Level Averages	
At 5 docs	0.0044
At 10 docs	0.0022
At 15 docs	0.0015
At 20 docs	0.0011
At 30 docs	0.0007
At 100 docs	0.0009
At 200 docs	0.0007
At 500 docs	0.0006
At 1000 docs	0.0005
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0004



Summary Statistics	
Run Number	DCU96C-category B, manual
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel.ret:	25

Recall Level Precision Averages	
Recall	Precision
0.00	0.0149
0.10	0.0001
0.20	0.0000
0.30	0.0000
0.40	0.0000
0.50	0.0000
0.60	0.0000
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0003

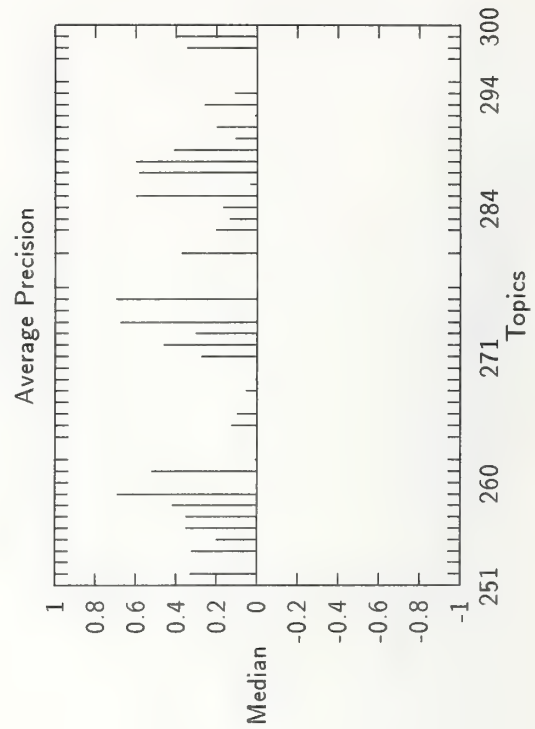
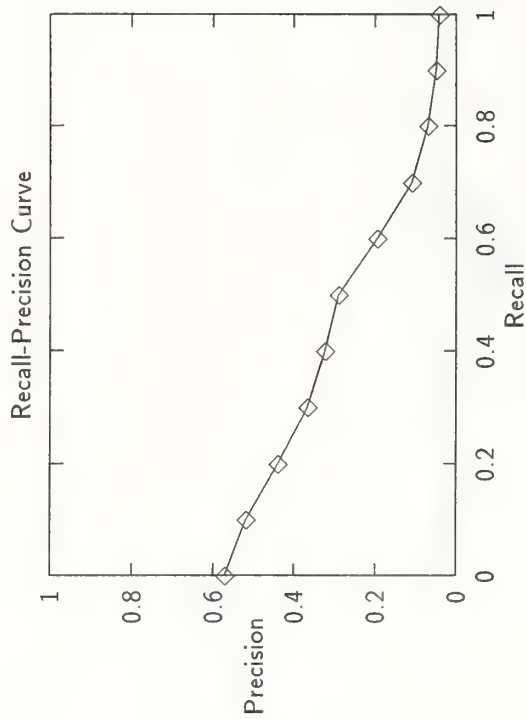
Document Level Averages	
	Precision
At 5 docs	0.0044
At 10 docs	0.0022
At 15 docs	0.0015
At 20 docs	0.0033
At 30 docs	0.0022
At 100 docs	0.0018
At 200 docs	0.0012
At 500 docs	0.0008
At 1000 docs	0.0006
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0010



Summary Statistics	
Run Number	uncisl-category B, manual
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	651

Recall Level Precision Averages	
Recall	Precision
0.00	0.5704
0.10	0.5185
0.20	0.4403
0.30	0.3674
0.40	0.3242
0.50	0.2904
0.60	0.1961
0.70	0.1092
0.80	0.0706
0.90	0.0493
1.00	0.0408
Average precision over all relevant docs	
non-interpolated	0.2510

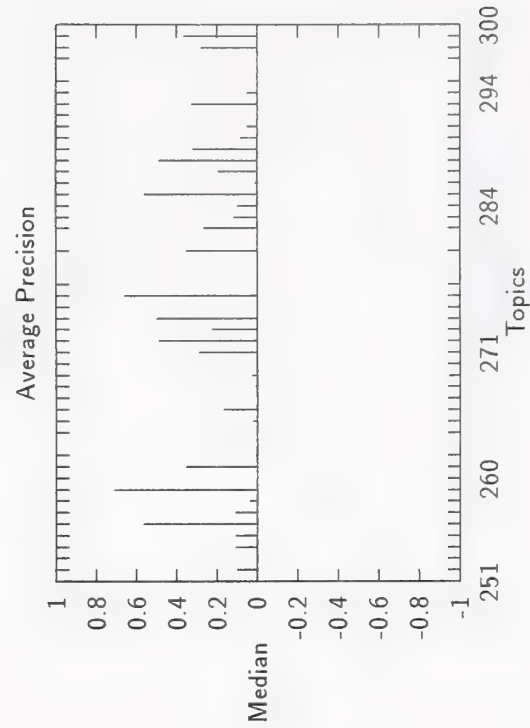
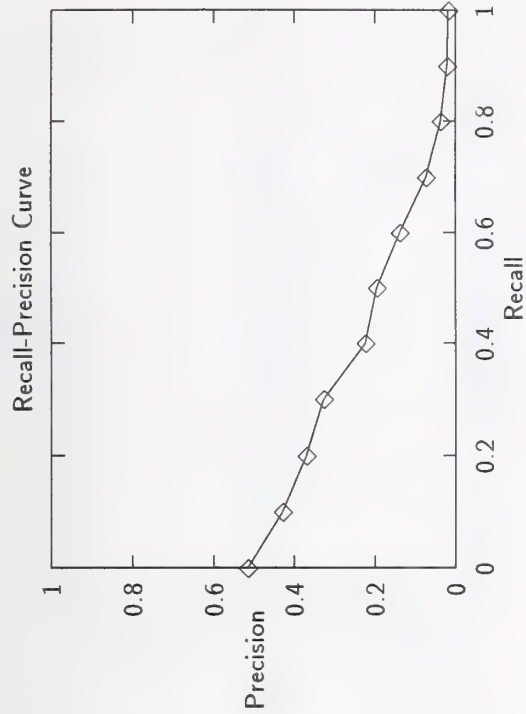
Document Level Averages	
At 5 docs	0.3867
At 10 docs	0.3289
At 15 docs	0.2785
At 20 docs	0.2356
At 30 docs	0.1807
At 100 docs	0.0824
At 200 docs	0.0488
At 500 docs	0.0249
At 1000 docs	0.0145
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2470



Summary Statistics		
Run Number	uncis2-category B, manual	45
Total number of documents over all topics		
Retrieved:	45000	
Relevant:	1064	
Rel_ret:	607	

Recall Level Precision Averages	
Recall	Precision
0.00	0.5156
0.10	0.4299
0.20	0.3703
0.30	0.3270
0.40	0.2248
0.50	0.1957
0.60	0.1389
0.70	0.0730
0.80	0.0371
0.90	0.0209
1.00	0.0181
Average precision over all relevant docs	
non-interpolated	0.1948

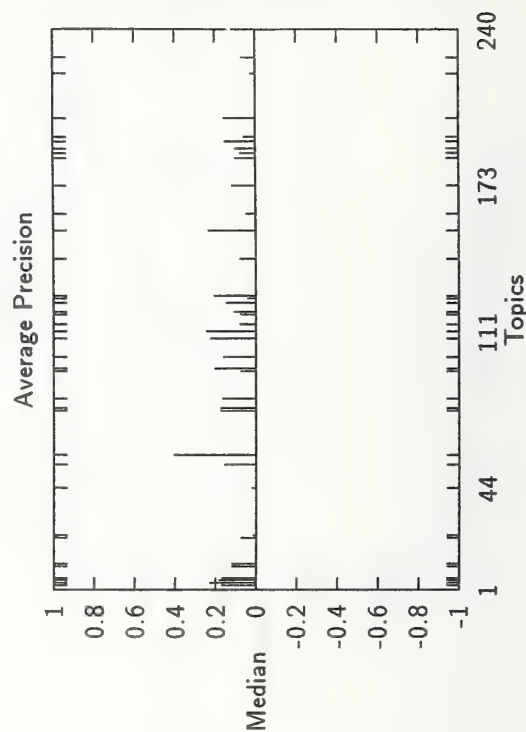
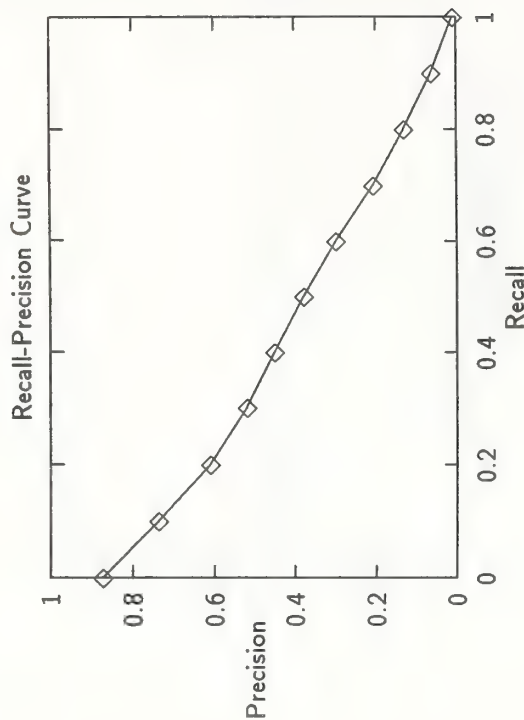
Document Level Averages	
	Precision
At 5 docs	0.3333
At 10 docs	0.2844
At 15 docs	0.2281
At 20 docs	0.1956
At 30 docs	0.1459
At 100 docs	0.0644
At 200 docs	0.0407
At 500 docs	0.0215
At 1000 docs	0.0135
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2123



Summary Statistics	
Run Number	city96r1-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	4167

Recall Level Precision Averages	
Recall	Precision
0.00	0.8723
0.10	0.7374
0.20	0.6104
0.30	0.5198
0.40	0.4518
0.50	0.3806
0.60	0.3004
0.70	0.2081
0.80	0.1326
0.90	0.0641
1.00	0.0089
Average precision over all relevant docs	
non-interpolated	0.3721

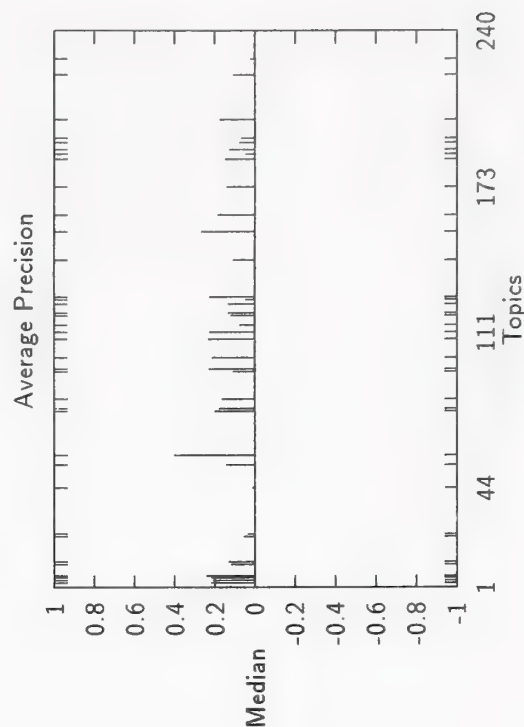
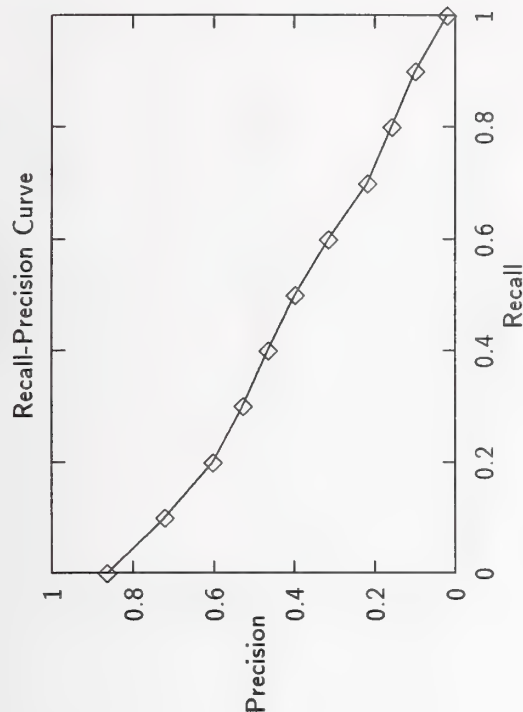
Document Level Averages	
At 5 docs	0.6923
At 10 docs	0.6462
At 15 docs	0.5880
At 20 docs	0.5462
At 30 docs	0.5085
At 100 docs	0.3710
At 200 docs	0.2814
At 500 docs	0.1689
At 1000 docs	0.1068
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4048



Summary Statistics	
Run Number	city96r2-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	4266

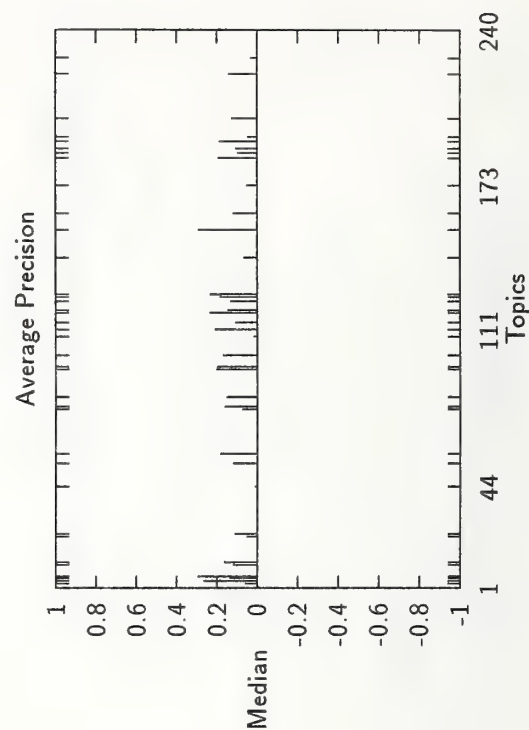
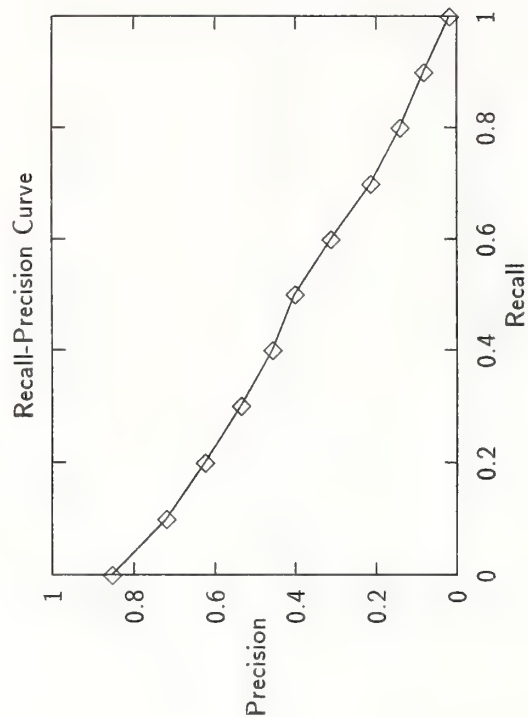
Recall Level Precision Averages	
Recall	Precision
0.00	0.8654
0.10	0.7222
0.20	0.6018
0.30	0.5274
0.40	0.4659
0.50	0.3993
0.60	0.3164
0.70	0.2189
0.80	0.1585
0.90	0.0996
1.00	0.0202
Average precision over all relevant docs	
non-interpolated	0.3860

Document Level Averages	
	Precision
At 5 docs	0.6615
At 10 docs	0.6462
At 15 docs	0.6085
At 20 docs	0.5756
At 30 docs	0.5316
At 100 docs	0.3800
At 200 docs	0.2933
At 500 docs	0.1756
At 1000 docs	0.1094
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4151



Summary Statistics	
Run Number	Cor5R1cc-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel _{ret} :	4240

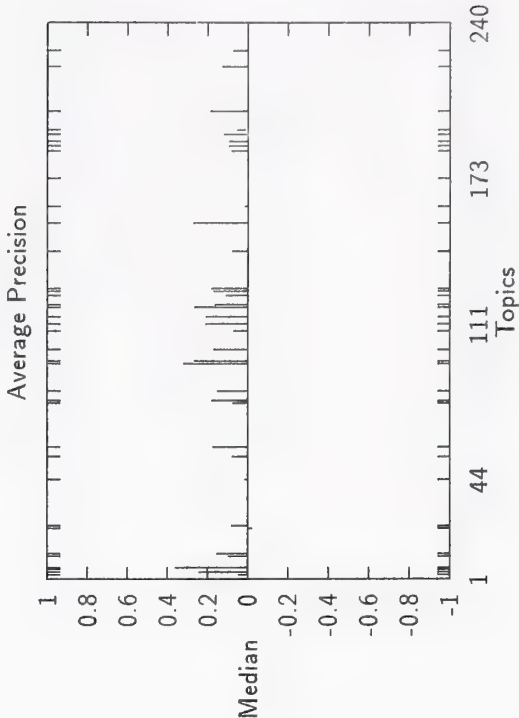
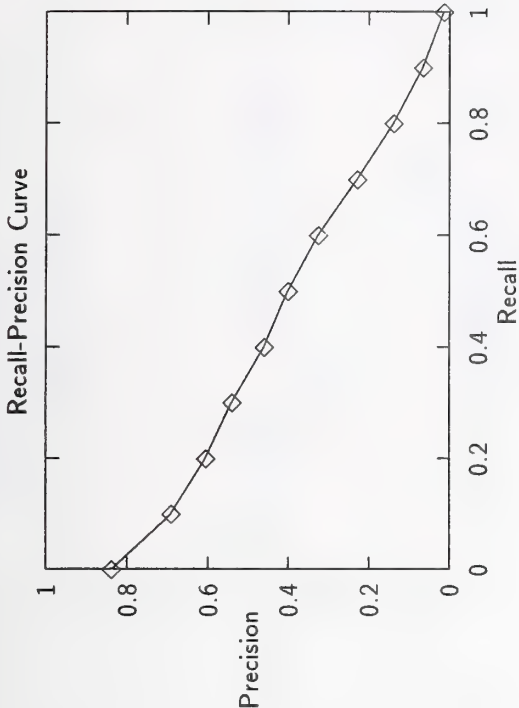
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.8541	At 5 docs	0.6359
0.10	0.7198	At 10 docs	0.5897
0.20	0.6243	At 15 docs	0.5709
0.30	0.5365	At 20 docs	0.5449
0.40	0.4576	At 30 docs	0.5094
0.50	0.4019	At 100 docs	0.3782
0.60	0.3138	At 200 docs	0.2910
0.70	0.2152	At 500 docs	0.1730
0.80	0.1421	At 1000 docs	0.1087
0.90	0.0834	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0189	Exact	0.4133
Average precision over all relevant docs			
non-interpolated	0.3807		



Summary Statistics	
Run Number	Cor5R2cr-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	4304

Recall Level Precision Averages	
Recall	Precision
0.00	0.8394
0.10	0.6921
0.20	0.6057
0.30	0.5403
0.40	0.4603
0.50	0.4025
0.60	0.3274
0.70	0.2306
0.80	0.1399
0.90	0.0668
1.00	0.0144
Average precision over all relevant docs	
non-interpolated	0.3759

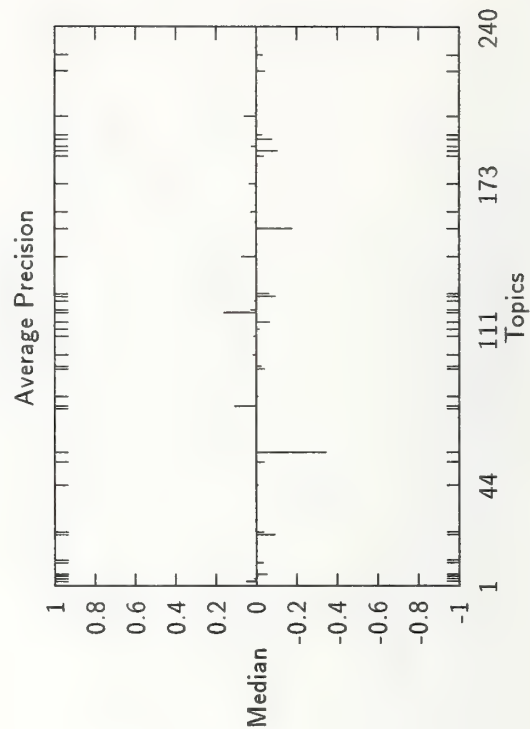
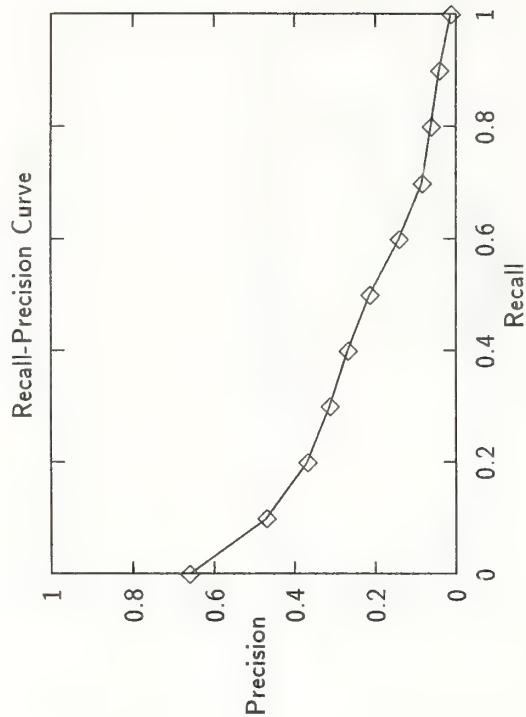
Document Level Averages	
	Precision
At 5 docs	0.6205
At 10 docs	0.5974
At 15 docs	0.5744
At 20 docs	0.5487
At 30 docs	0.5205
At 100 docs	0.3921
At 200 docs	0.2909
At 500 docs	0.1736
At 1000 docs	0.1104
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4133



Summary Statistics	
Run Number	genrl5-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	3560

Recall Level Precision Averages	
Recall	Precision
0.00	0.6612
0.10	0.4698
0.20	0.3690
0.30	0.3129
0.40	0.2679
0.50	0.2146
0.60	0.1416
0.70	0.0837
0.80	0.0625
0.90	0.0412
1.00	0.0120
Average precision over all relevant docs	
non-interpolated	0.2157

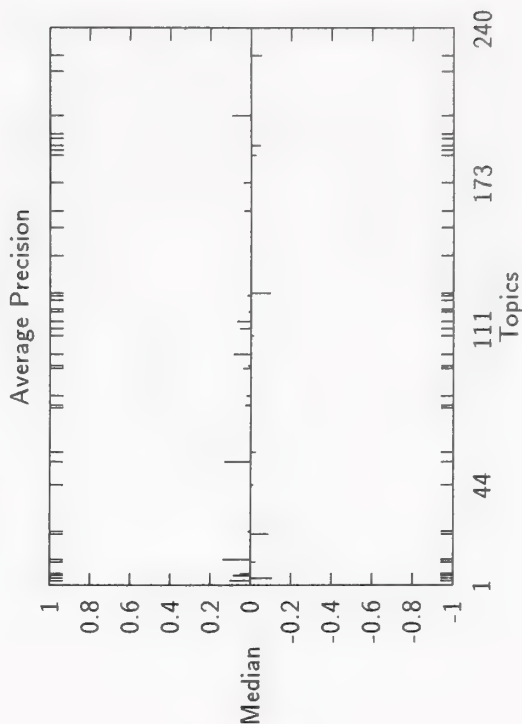
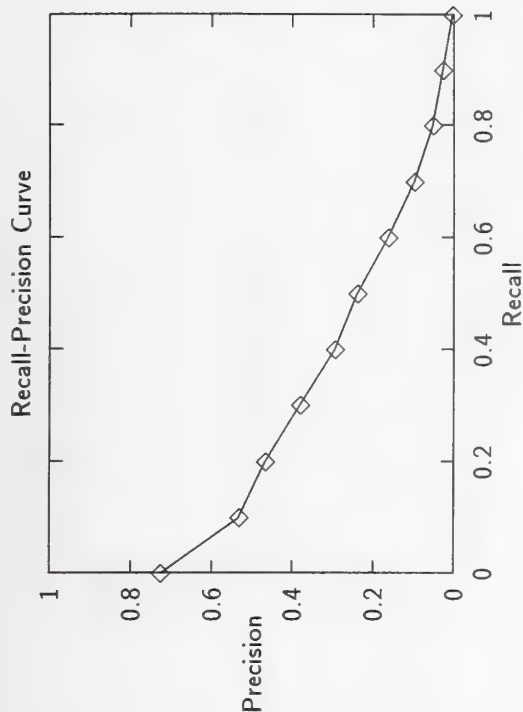
Document Level Averages	
	Precision
At 5 docs	0.4513
At 10 docs	0.4103
At 15 docs	0.3880
At 20 docs	0.3718
At 30 docs	0.3436
At 100 docs	0.2869
At 200 docs	0.2292
At 500 docs	0.1475
At 1000 docs	0.0913
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2700



Summary Statistics	
Run Number	genrl6-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel.Ret:	3613

Recall Level Precision Averages	
Recall	Precision
0.00	0.7273
0.10	0.5325
0.20	0.4677
0.30	0.3809
0.40	0.2951
0.50	0.2399
0.60	0.1639
0.70	0.0983
0.80	0.0528
0.90	0.0278
1.00	0.0030
Average precision over all relevant docs	
non-interpolated	0.2498

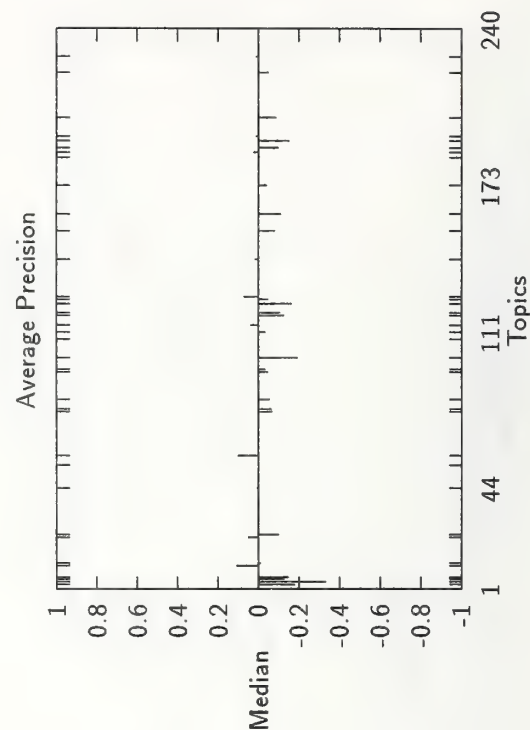
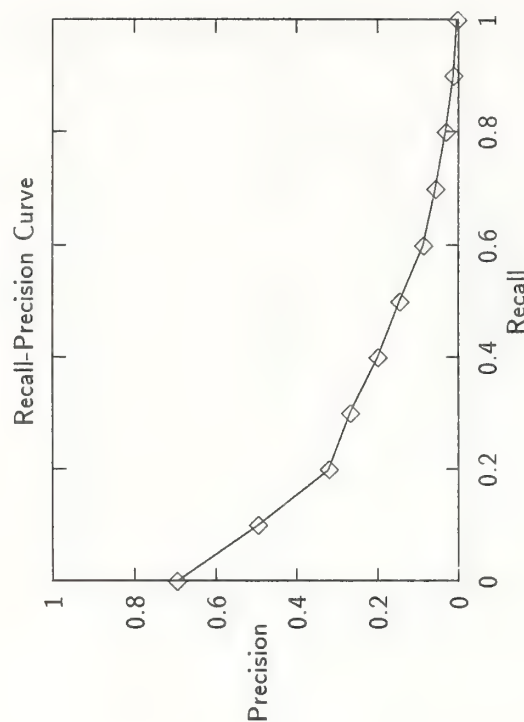
Document Level Averages	
	Precision
At 5 docs	0.5487
At 10 docs	0.4974
At 15 docs	0.4718
At 20 docs	0.4385
At 30 docs	0.4026
At 100 docs	0.3128
At 200 docs	0.2296
At 500 docs	0.1410
At 1000 docs	0.0926
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3014



Summary Statistics	
Run Number	itidp1-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	3211

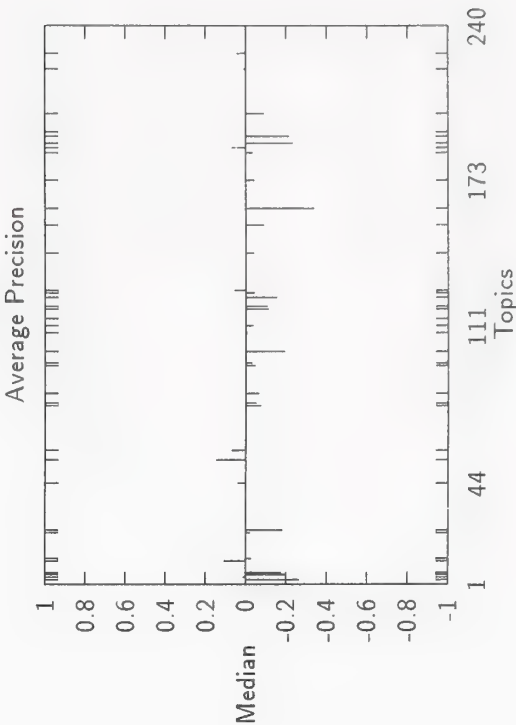
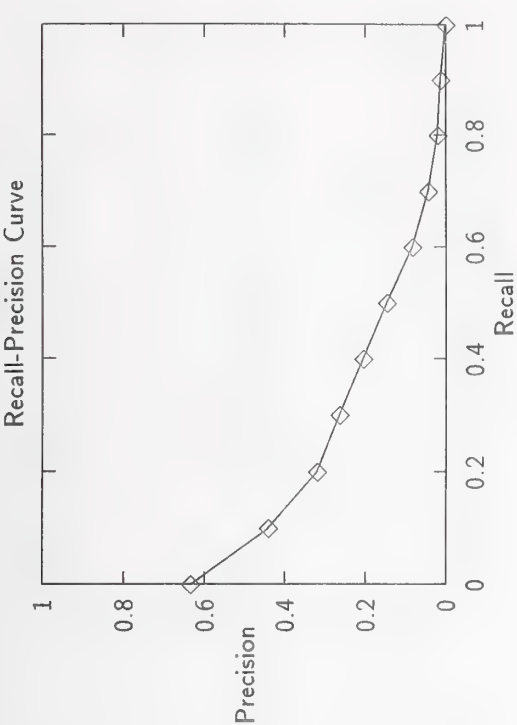
Recall Level Precision Averages	
Recall	Precision
0.00	0.6956
0.10	0.4955
0.20	0.3215
0.30	0.2679
0.40	0.2004
0.50	0.1462
0.60	0.0874
0.70	0.0574
0.80	0.0317
0.90	0.0134
1.00	0.0024
Average precision over all relevant docs	
non-interpolated	0.1866

Document Level Averages	
	Precision
At 5 docs	0.4615
At 10 docs	0.4282
At 15 docs	0.3932
At 20 docs	0.3628
At 30 docs	0.3325
At 100 docs	0.2382
At 200 docs	0.1824
At 500 docs	0.1190
At 1000 docs	0.0823
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2388



Summary Statistics	
Run Number	itidp2-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	2946

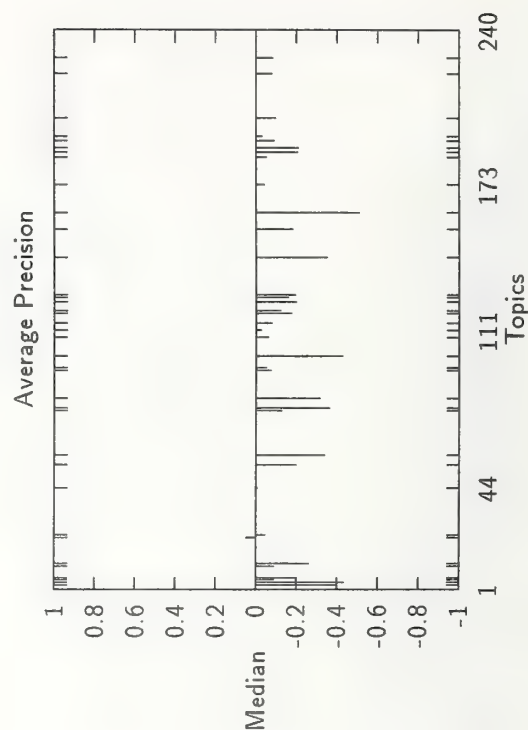
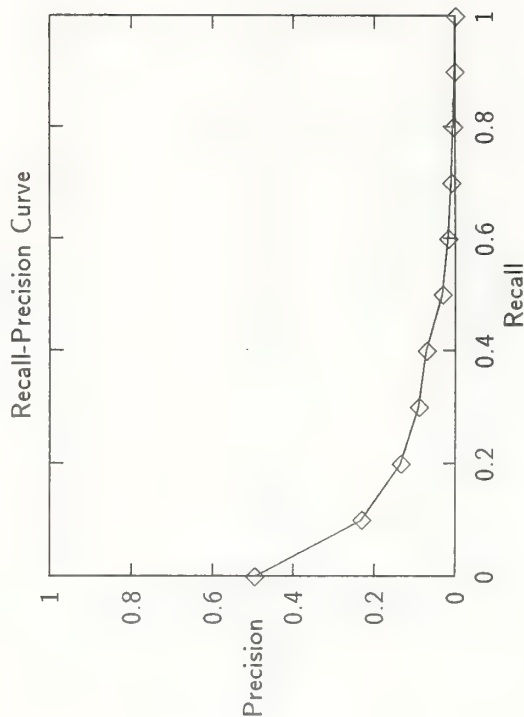
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.6342	At 5 docs	0.4564
0.10	0.4424	At 10 docs	0.4103
0.20	0.3217	At 15 docs	0.3744
0.30	0.2658	At 20 docs	0.3564
0.40	0.2069	At 30 docs	0.3274
0.50	0.1481	At 100 docs	0.2292
0.60	0.0838	At 200 docs	0.1740
0.70	0.0452	At 500 docs	0.1114
0.80	0.0206	At 1000 docs	0.0755
0.90	0.0128	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0006	Exact	0.2310
Average precision over all relevant docs			
non-interpolated	0.1784		



Summary Statistics	
Run Number	nmsu-1-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	2102

Recall Level Precision Averages	
Recall	Precision
0.00	0.4967
0.10	0.2318
0.20	0.1356
0.30	0.0900
0.40	0.0713
0.50	0.0315
0.60	0.0182
0.70	0.0095
0.80	0.0043
0.90	0.0023
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0716

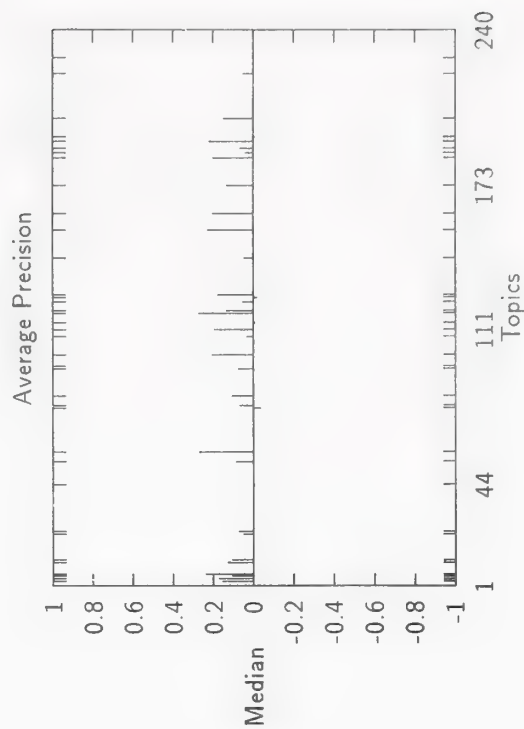
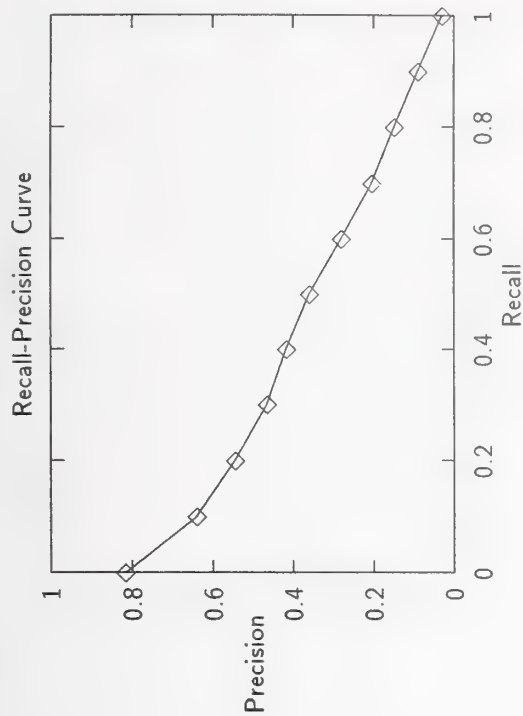
Document Level Averages	
At 5 docs	0.2359
At 10 docs	0.2051
At 15 docs	0.1829
At 20 docs	0.1718
At 30 docs	0.1641
At 100 docs	0.1341
At 200 docs	0.1082
At 500 docs	0.0747
At 1000 docs	0.0539
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1218



Summary Statistics	
Run Number	pircsg0-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	4105

Recall Level Precision Averages	
Recall	Precision
0.00	0.8175
0.10	0.6410
0.20	0.5458
0.30	0.4652
0.40	0.4175
0.50	0.3611
0.60	0.2821
0.70	0.2046
0.80	0.1487
0.90	0.0904
1.00	0.0308
Average precision over all relevant docs	
non-interpolated	0.3477

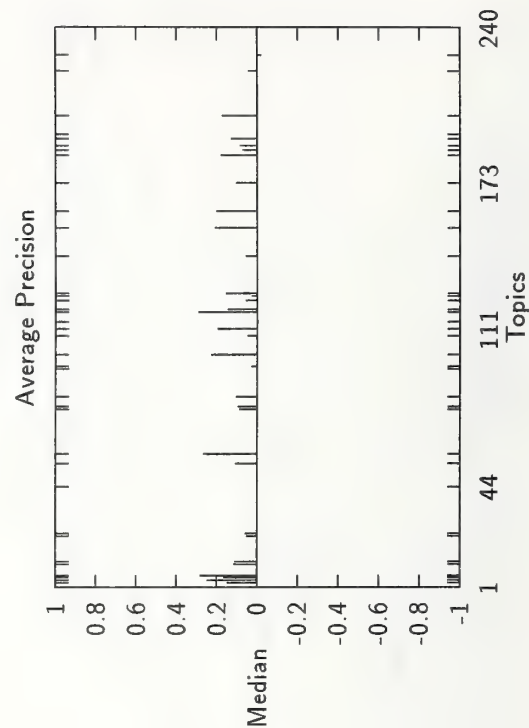
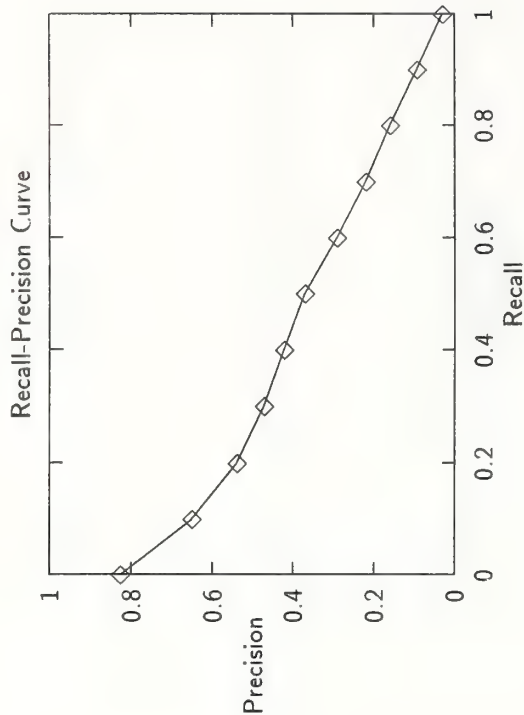
Document Level Averages	
	Precision
At 5 docs	0.5949
At 10 docs	0.5615
At 15 docs	0.5419
At 20 docs	0.5154
At 30 docs	0.4795
At 100 docs	0.3713
At 200 docs	0.2801
At 500 docs	0.1689
At 1000 docs	0.1053
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3735



Summary Statistics	
Run Number	pircsg6-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
ReRet:	4182

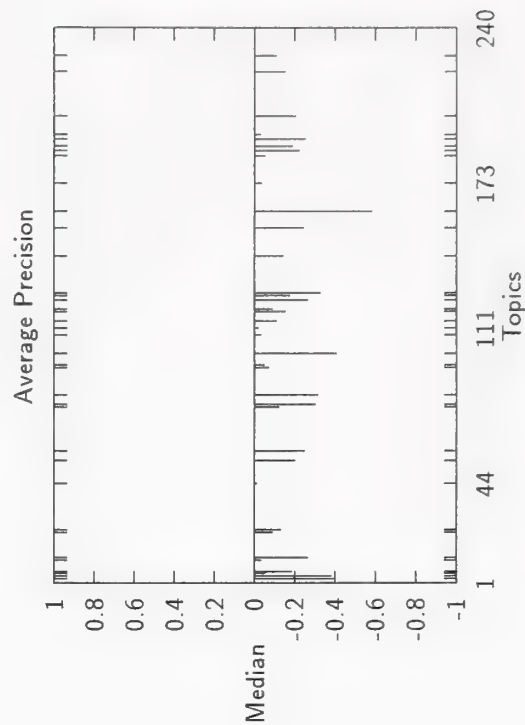
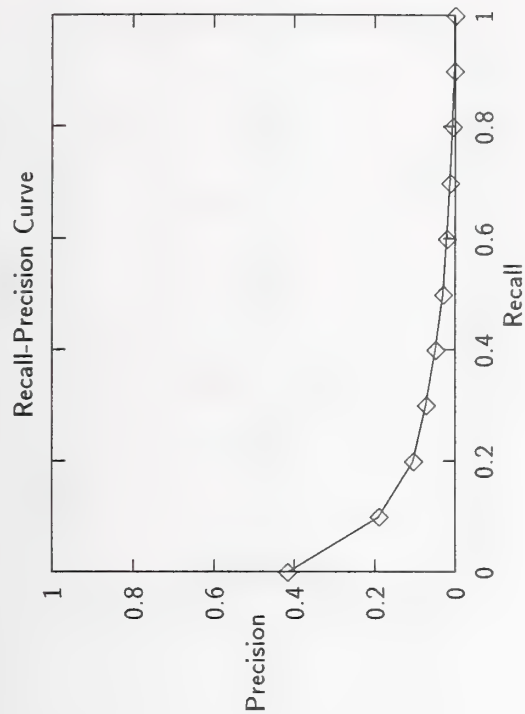
Recall Level Precision Averages	
Recall	Precision
0.00	0.8274
0.10	0.6497
0.20	0.5396
0.30	0.4725
0.40	0.4235
0.50	0.3723
0.60	0.2928
0.70	0.2214
0.80	0.1602
0.90	0.0939
1.00	0.0301
Average precision over all relevant docs	
non-interpolated	0.3560

Document Level Averages	
	Precision
At 5 docs	0.6000
At 10 docs	0.5538
At 15 docs	0.5385
At 20 docs	0.5154
At 30 docs	0.4889
At 100 docs	0.3762
At 200 docs	0.2849
At 500 docs	0.1708
At 1000 docs	0.1072
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3796



Summary Statistics	
Run Number	rutAPglob-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	2241

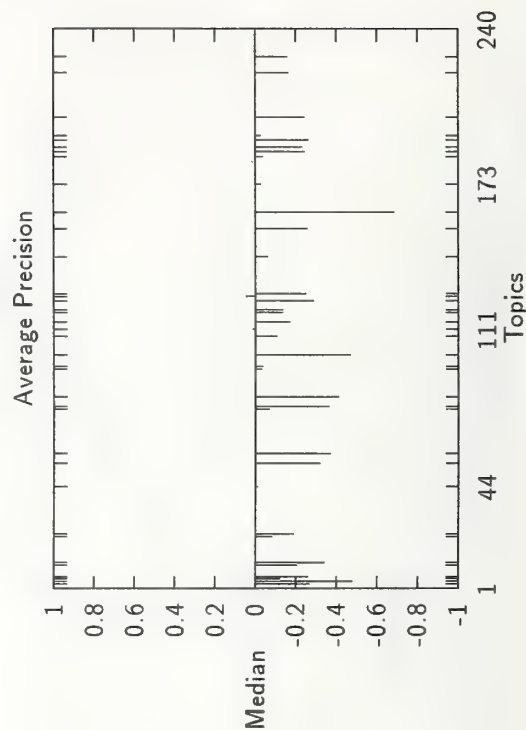
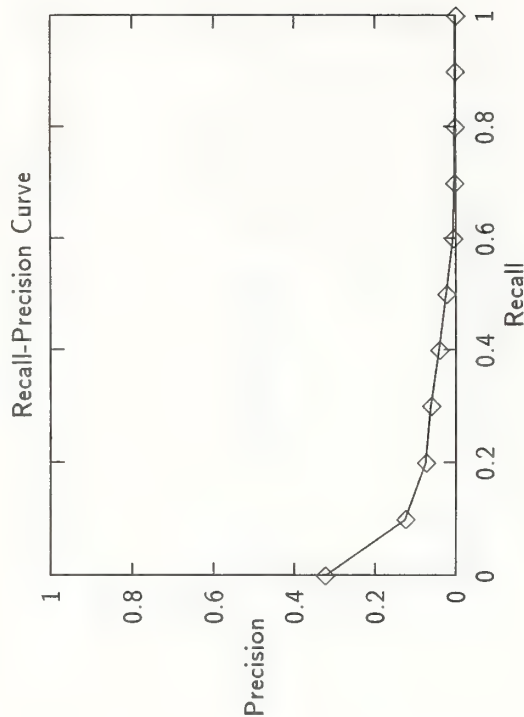
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.4184	At 5 docs	0.2615
0.10	0.1914	At 10 docs	0.2308
0.20	0.1042	At 15 docs	0.1932
0.30	0.0724	At 20 docs	0.1936
0.40	0.0499	At 30 docs	0.1726
0.50	0.0310	At 100 docs	0.1279
0.60	0.0213	At 200 docs	0.1069
0.70	0.0124	At 500 docs	0.0748
0.80	0.0064	At 1000 docs	0.0575
0.90	0.0012	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0000	Exact	0.1009
Average precision over all relevant docs			
non-interpolated	0.0621		



Summary Statistics	
Run Number	rutAPspt-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	1458

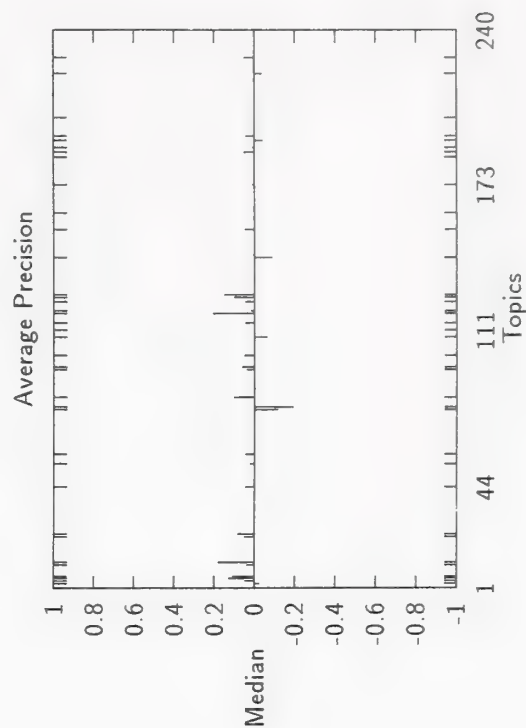
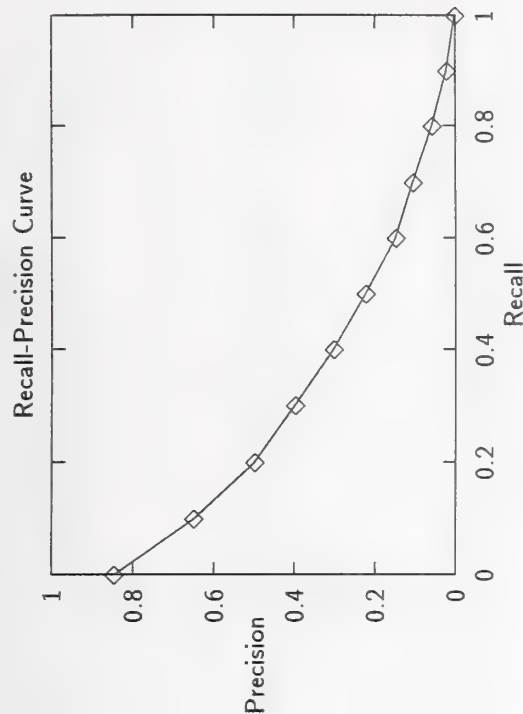
Recall Level Precision Averages	
Recall	Precision
0.00	0.3248
0.10	0.1253
0.20	0.0754
0.30	0.0618
0.40	0.0419
0.50	0.0246
0.60	0.0066
0.70	0.0030
0.80	0.0013
0.90	0.0010
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0420

Document Level Averages	
	Precision
At 5 docs	0.1487
At 10 docs	0.1205
At 15 docs	0.1128
At 20 docs	0.1090
At 30 docs	0.1000
At 100 docs	0.0903
At 200 docs	0.0714
At 500 docs	0.0504
At 1000 docs	0.0374
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0689



Summary Statistics	
Run Number	ETHru1-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
RelRet:	3160

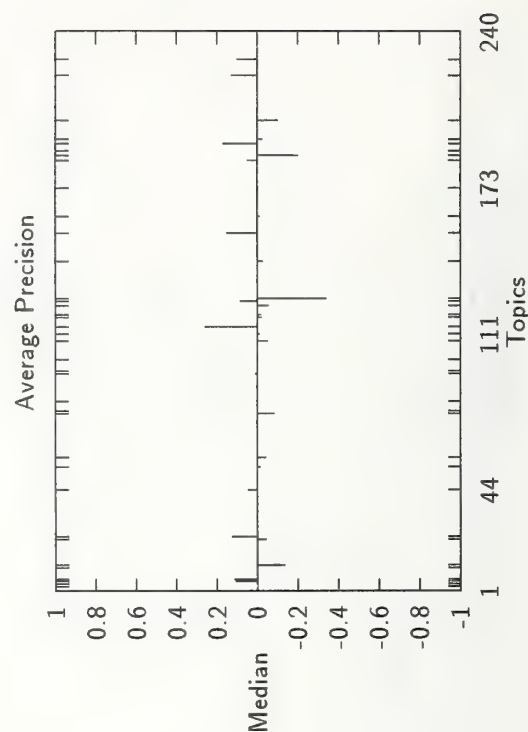
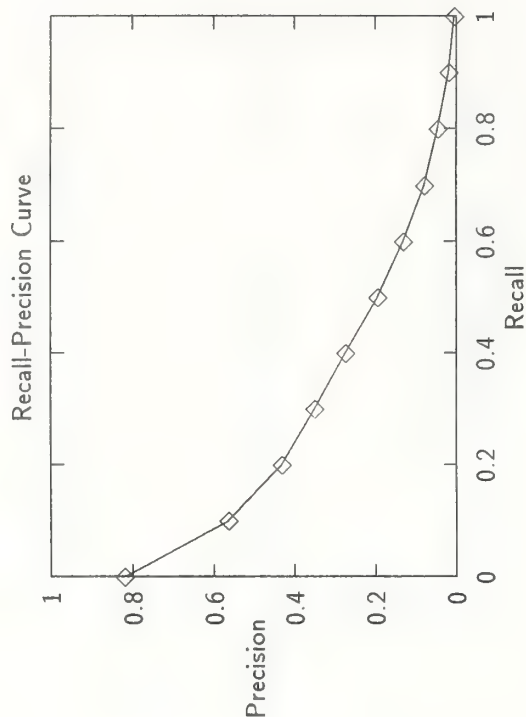
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.8474	At 5 docs	0.5744
0.10	0.6510	At 10 docs	0.5179
0.20	0.4986	At 15 docs	0.4752
0.30	0.3996	At 20 docs	0.4615
0.40	0.3036	At 30 docs	0.4462
0.50	0.2241	At 100 docs	0.3303
0.60	0.1491	At 200 docs	0.2488
0.70	0.1062	At 500 docs	0.1437
0.80	0.0583	At 1000 docs	0.0810
0.90	0.0216	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0013	Exact	0.3243
Average precision over all relevant docs			
non-interpolated	0.2704		



Summary Statistics	
Run Number	brkly13-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	2990

Recall Level Precision Averages	
Recall	Precision
0.00	0.8208
0.10	0.5650
0.20	0.4331
0.30	0.3514
0.40	0.2754
0.50	0.1952
0.60	0.1322
0.70	0.0792
0.80	0.0458
0.90	0.0187
1.00	0.0048
Average precision over all relevant docs	
non-interpolated	0.2404

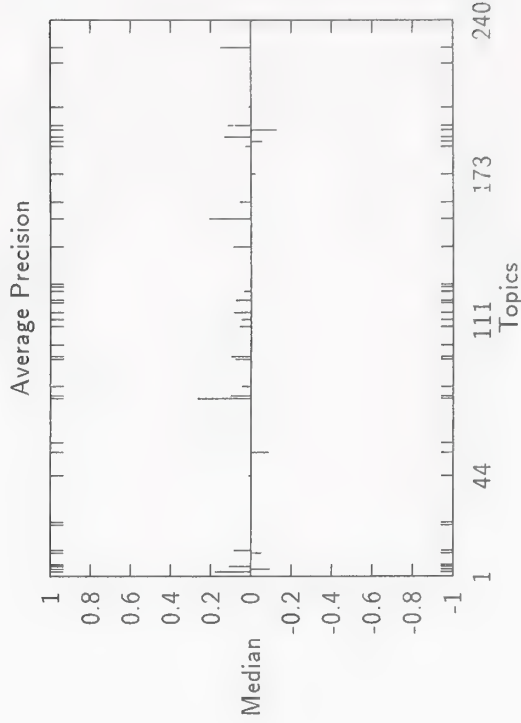
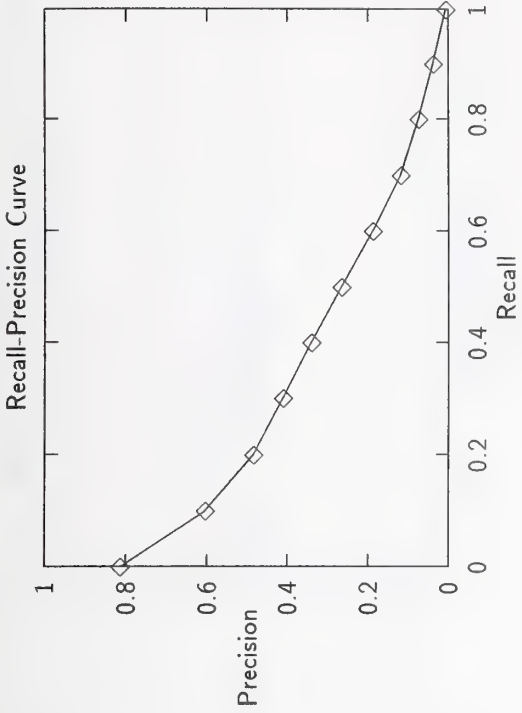
Document Level Averages	
	Precision
At 5 docs	0.5641
At 10 docs	0.4949
At 15 docs	0.4803
At 20 docs	0.4487
At 30 docs	0.4085
At 100 docs	0.3205
At 200 docs	0.2128
At 500 docs	0.1203
At 1000 docs	0.0767
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2981



Summary Statistics	
Run Number	brkly14-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	3791

Recall Level Precision Averages	
Recall	Precision
0.00	0.8138
0.10	0.6045
0.20	0.4844
0.30	0.4121
0.40	0.3415
0.50	0.2666
0.60	0.1885
0.70	0.1190
0.80	0.0754
0.90	0.0387
1.00	0.0056
Average precision over all relevant docs	
non-interpolated	0.2795

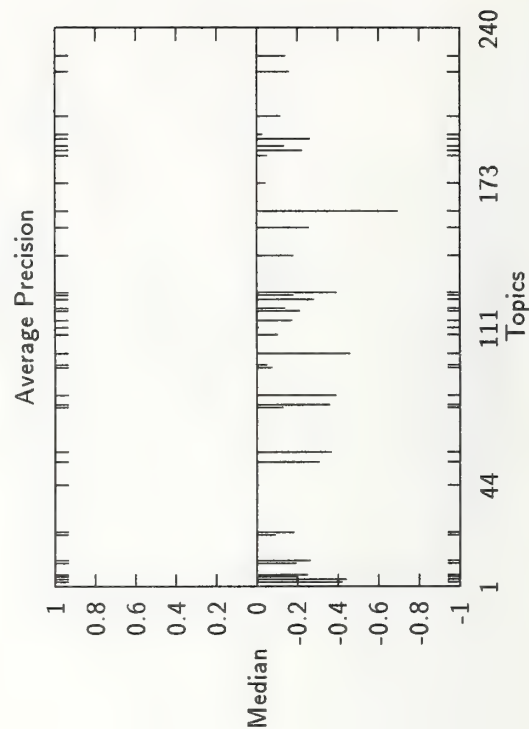
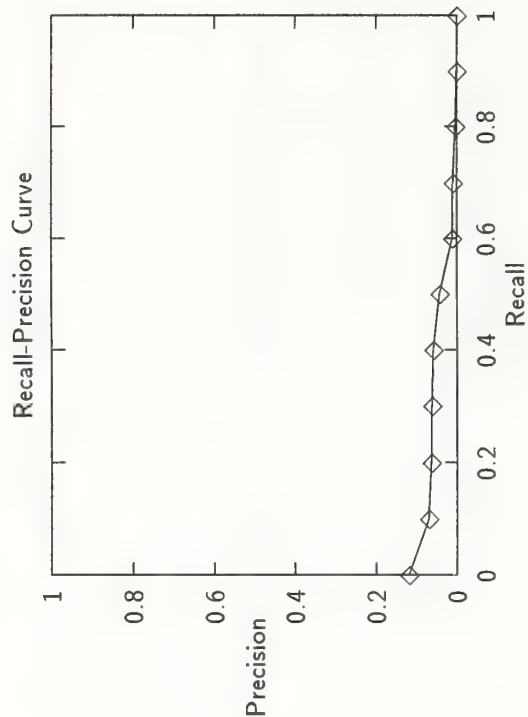
Document Level Averages	
At 5 docs	0.5385
At 10 docs	0.5103
At 15 docs	0.4889
At 20 docs	0.4590
At 30 docs	0.4291
At 100 docs	0.3249
At 200 docs	0.2412
At 500 docs	0.1515
At 1000 docs	0.0972
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3357



Summary Statistics	
Run Number	ispaR-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	2563

Recall Level Precision Averages	
Recall	Precision
0.00	0.1197
0.10	0.0695
0.20	0.0636
0.30	0.0616
0.40	0.0585
0.50	0.0428
0.60	0.0109
0.70	0.0089
0.80	0.0033
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0291

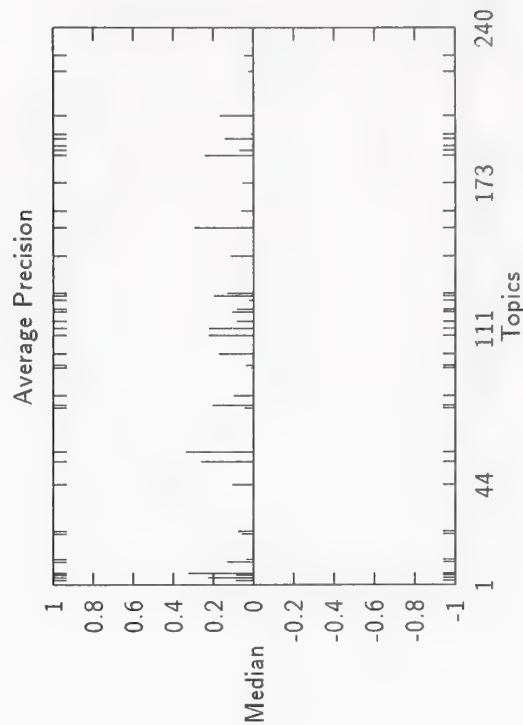
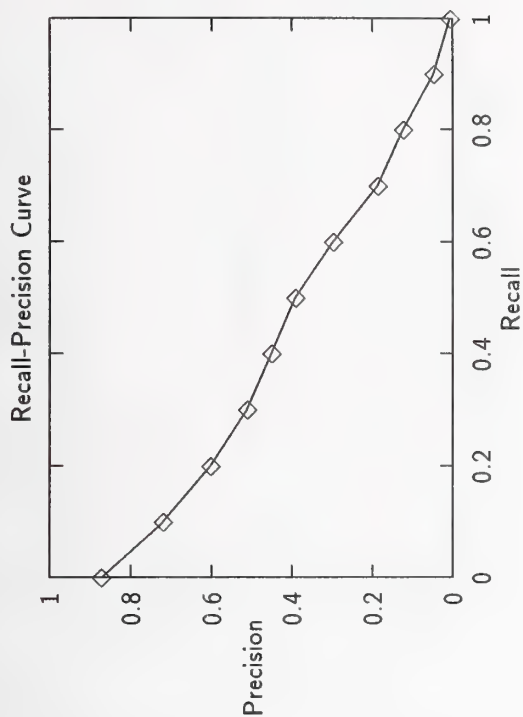
Document Level Averages	
	Precision
At 5 docs	0.0462
At 10 docs	0.0564
At 15 docs	0.0632
At 20 docs	0.0628
At 30 docs	0.0658
At 100 docs	0.0672
At 200 docs	0.0490
At 500 docs	0.0406
At 1000 docs	0.0657
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0443



Summary Statistics	
Run Number	INQ303—category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel.ret:	4084

Recall Level Precision Averages	
Recall	Precision
0.00	0.8726
0.10	0.7195
0.20	0.6031
0.30	0.5121
0.40	0.4508
0.50	0.3925
0.60	0.2985
0.70	0.1872
0.80	0.1247
0.90	0.0474
1.00	0.0068
Average precision over all relevant docs	
non-interpolated	0.3633

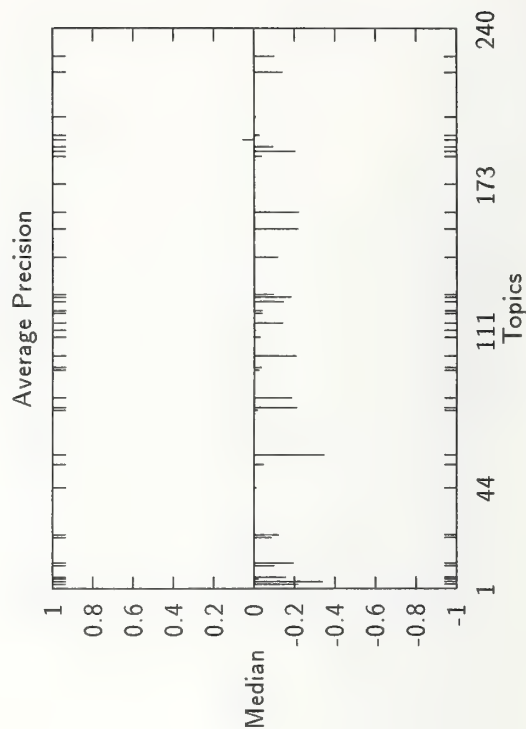
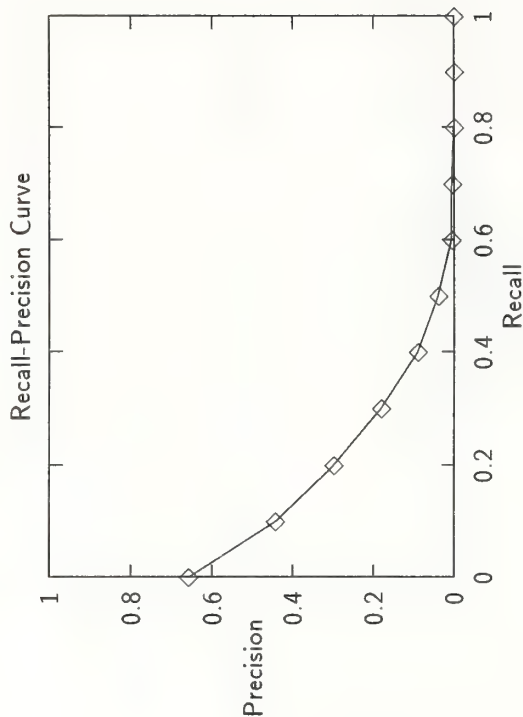
Document Level Averages	
	Precision
At 5 docs	0.5949
At 10 docs	0.5872
At 15 docs	0.5744
At 20 docs	0.5513
At 30 docs	0.5034
At 100 docs	0.3756
At 200 docs	0.2723
At 500 docs	0.1633
At 1000 docs	0.1047
R—Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3966



Summary Statistics	
Run Number	xerox.rout1-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	2037

Recall Level Precision Averages	
Recall	Precision
0.00	0.6578
0.10	0.4428
0.20	0.2986
0.30	0.1819
0.40	0.0914
0.50	0.0405
0.60	0.0066
0.70	0.0053
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.1286

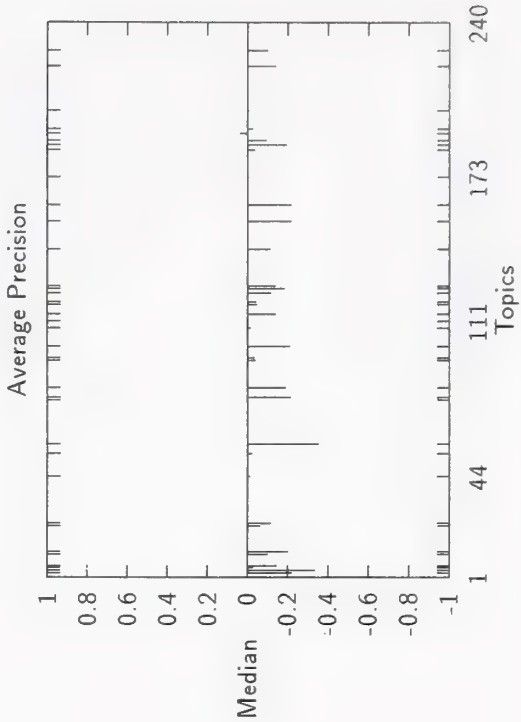
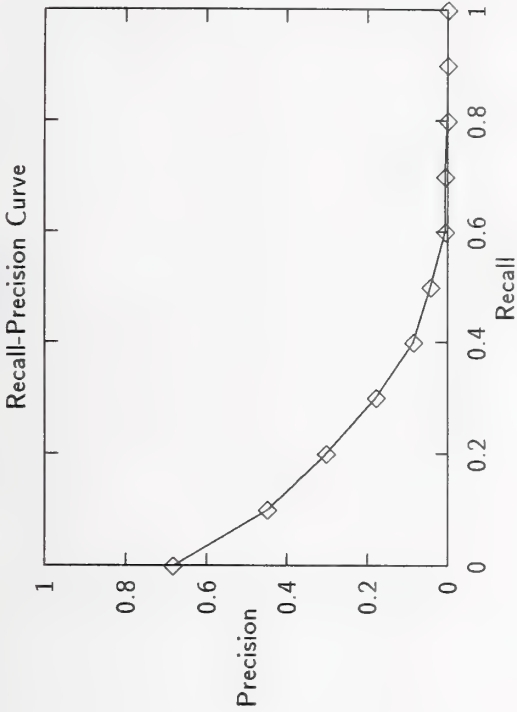
Document Level Averages	
At 5 docs	0.4974
At 10 docs	0.4487
At 15 docs	0.3966
At 20 docs	0.3628
At 30 docs	0.3350
At 100 docs	0.2313
At 200 docs	0.1565
At 500 docs	0.0846
At 1000 docs	0.0522
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1971



Summary Statistics		
Run Number	xerox.rout2-category A, automatic	
Number of Topics		39
Total number of documents over all topics		
Retrieved:		39000
Relevant:		5799
Rel.ret:		2037

Recall Level Precision Averages	
Recall	Precision
0.00	0.6840
0.10	0.4500
0.20	0.3038
0.30	0.1804
0.40	0.0855
0.50	0.0424
0.60	0.0069
0.70	0.0058
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.1303

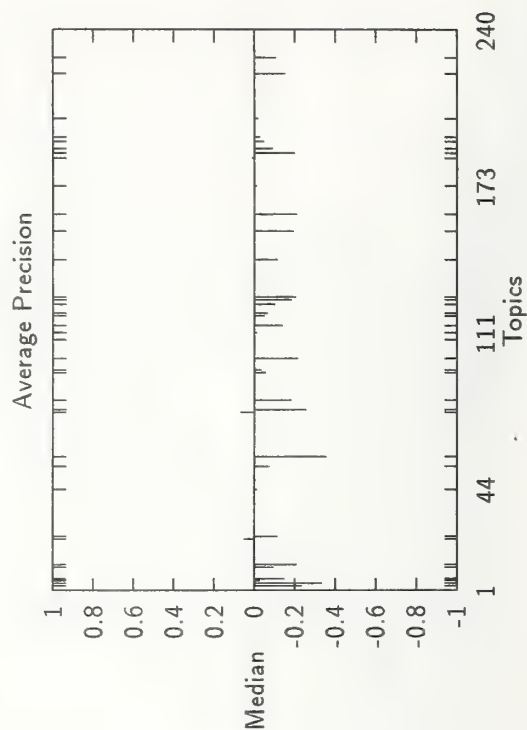
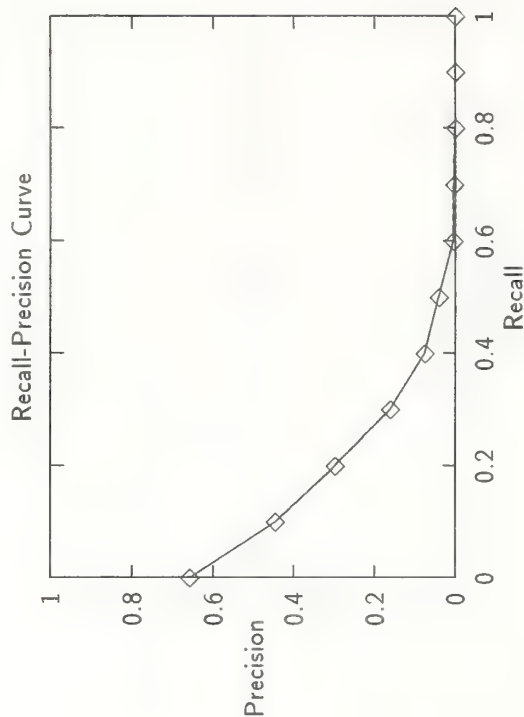
Document Level Averages	
	Precision
At 5 docs	0.5128
At 10 docs	0.4667
At 15 docs	0.4034
At 20 docs	0.3782
At 30 docs	0.3368
At 100 docs	0.2351
At 200 docs	0.1597
At 500 docs	0.0847
At 1000 docs	0.0522
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2038



Summary Statistics	
Run Number	xerox.rout3-category A, automatic
Number of Topics	39
Total number of documents over all topics	
Retrieved:	39000
Relevant:	5799
Rel_ret:	2037

Recall Level Precision Averages	
Recall	Precision
0.00	0.6577
0.10	0.4461
0.20	0.2988
0.30	0.1616
0.40	0.0763
0.50	0.0417
0.60	0.0041
0.70	0.0032
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.1284

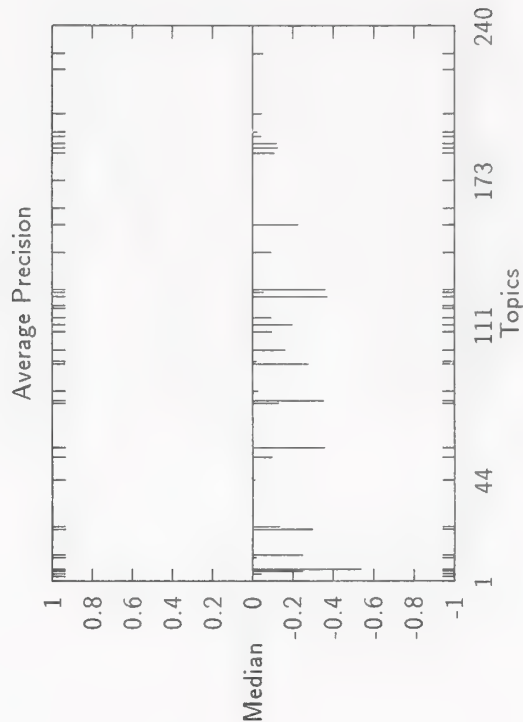
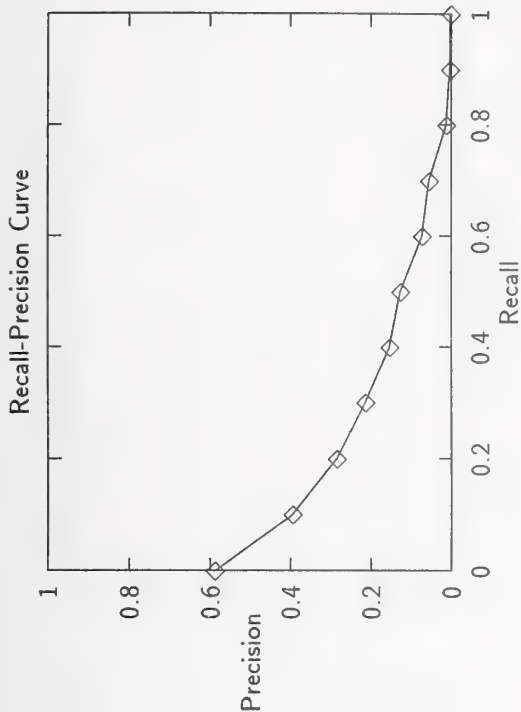
Document Level Averages	
	Precision
At 5 docs	0.4667
At 10 docs	0.4462
At 15 docs	0.4017
At 20 docs	0.3782
At 30 docs	0.3462
At 100 docs	0.2300
At 200 docs	0.1554
At 500 docs	0.0846
At 1000 docs	0.0522
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1992



Summary Statistics	
Run Number	erlR1-category A, manual
Number of Topics	39
Total number of documents over all topics	
Retrieved:	38085
Relevant:	5799
Rel_ret:	2556

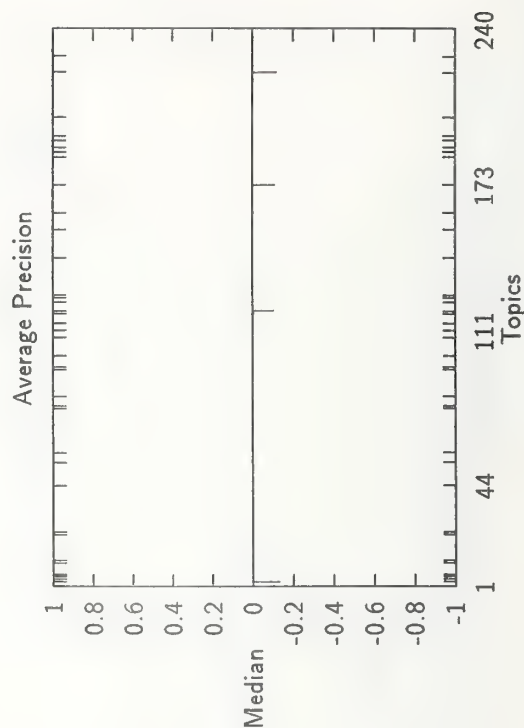
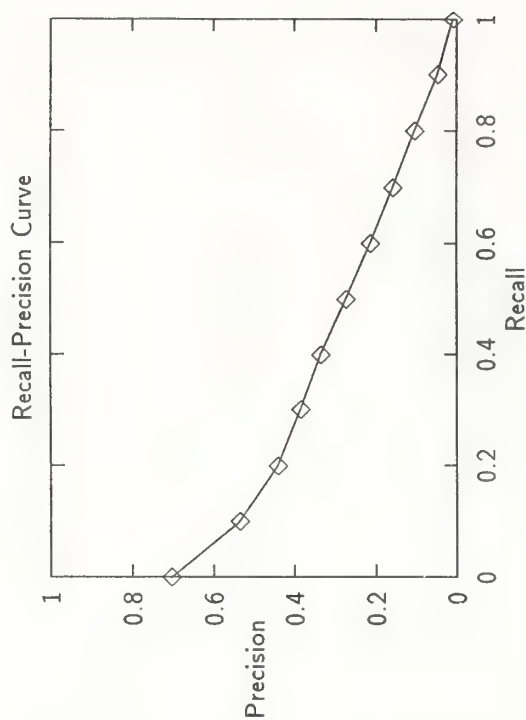
Recall Level Precision Averages	
Recall	Precision
0.00	0.5889
0.10	0.3960
0.20	0.2863
0.30	0.2139
0.40	0.1537
0.50	0.1278
0.60	0.0733
0.70	0.0554
0.80	0.0134
0.90	0.0027
1.00	0.0020
Average precision over all relevant docs	
non-interpolated	0.1473

Document Level Averages	
At 5 docs	0.3641
At 10 docs	0.3462
At 15 docs	0.3162
At 20 docs	0.3000
At 30 docs	0.2684
At 100 docs	0.2305
At 200 docs	0.1628
At 500 docs	0.0968
At 1000 docs	0.0655
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2115



Summary Statistics	
Run Number	uwgr0-category A, manual
Number of Topics	39
Total number of documents over all topics	
Retrieved:	28283
Relevant:	5799
Rel_ret:	3429

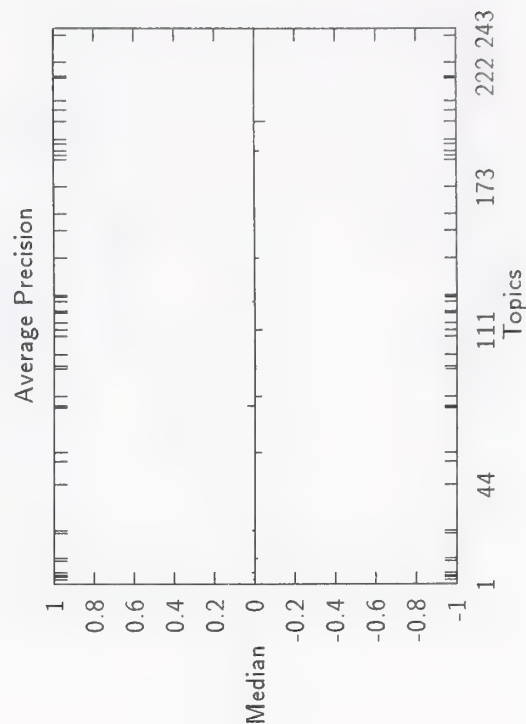
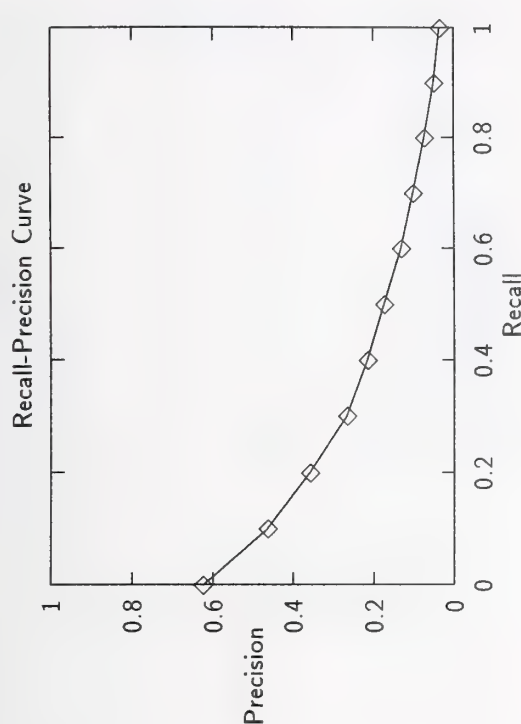
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.7050	At 5 docs	0.5077
0.10	0.5352	At 10 docs	0.4564
0.20	0.4416	At 15 docs	0.4274
0.30	0.3857	At 20 docs	0.4218
0.40	0.3364	At 30 docs	0.3966
0.50	0.2751	At 100 docs	0.3028
0.60	0.2150	At 200 docs	0.2262
0.70	0.1595	At 500 docs	0.1372
0.80	0.1049	At 1000 docs	0.0879
0.90	0.0473	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0096	Exact	0.3159
Average precision over all relevant docs			
non-interpolated	0.2712		



Summary Statistics	
Run Number	Mercure-r1-category B, automatic
Number of Topics	43
Total number of documents over all topics	
Retrieved:	43000
Relevant:	2689
Rel_ret:	1963

Recall Level Precision Averages	
Recall	Precision
0.00	0.6245
0.10	0.4637
0.20	0.3592
0.30	0.2677
0.40	0.2161
0.50	0.1745
0.60	0.1314
0.70	0.1022
0.80	0.0736
0.90	0.0494
1.00	0.0348
Average precision over all relevant docs	
non-interpolated	0.2054

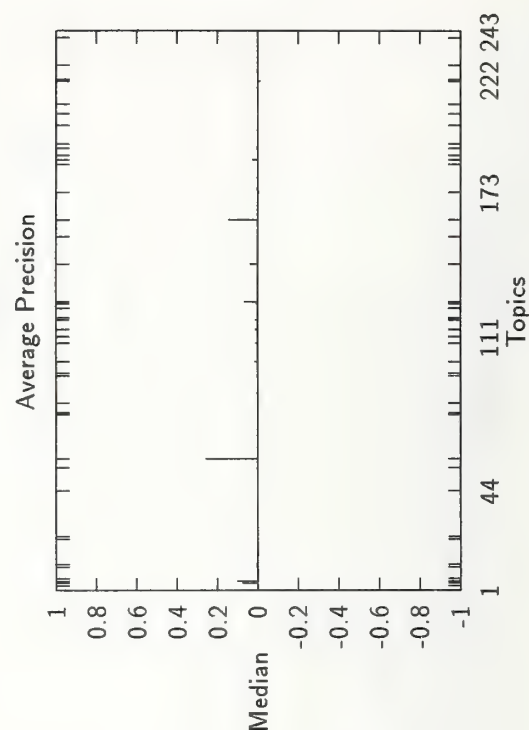
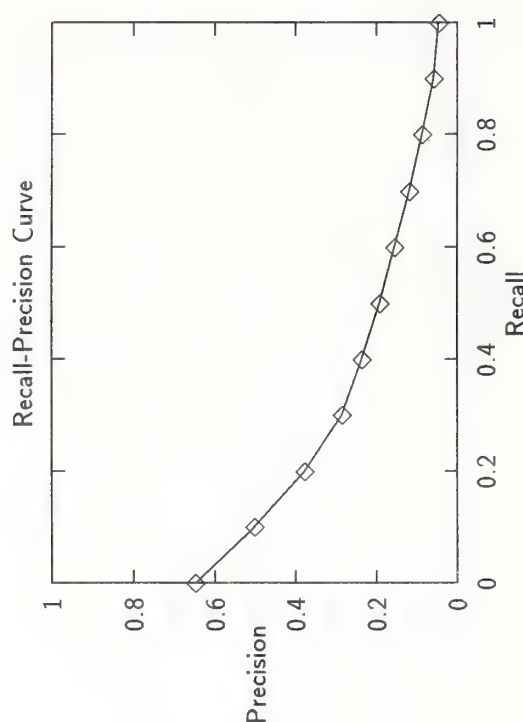
Document Level Averages	
	Precision
At 5 docs	0.3488
At 10 docs	0.3023
At 15 docs	0.2791
At 20 docs	0.2581
At 30 docs	0.2248
At 100 docs	0.1484
At 200 docs	0.1112
At 500 docs	0.0707
At 1000 docs	0.0457
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2297



Summary Statistics	
Run Number	Mercure-r2-category B, automatic
Number of Topics	43
Total number of documents over all topics	
Retrieved:	43000
Relevant:	2689
Rel _{ret} :	2031

Recall Level Precision Averages	
Recall	Precision
0.00	0.6489
0.10	0.5029
0.20	0.3779
0.30	0.2854
0.40	0.2364
0.50	0.1924
0.60	0.1553
0.70	0.1196
0.80	0.0878
0.90	0.0585
1.00	0.0464
Average precision over all relevant docs	
non-interpolated	0.2264

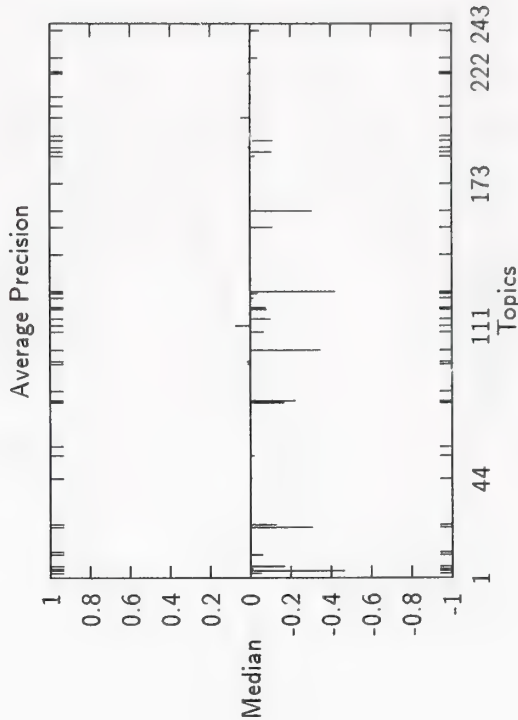
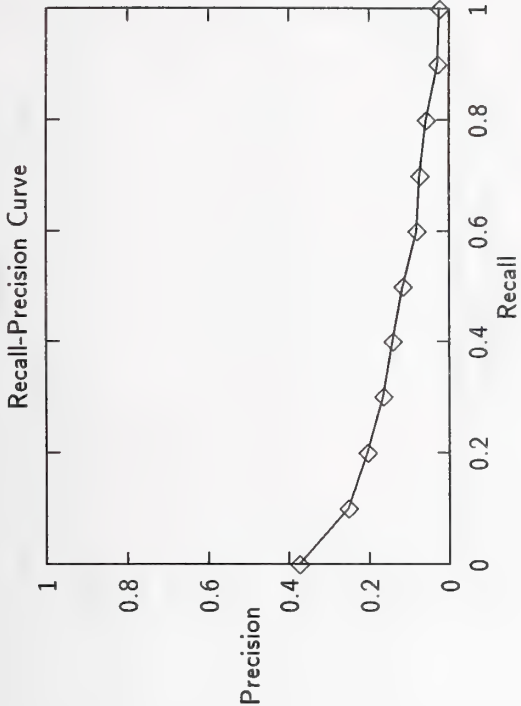
Document Level Averages	
At 5 docs	0.3767
At 10 docs	0.3256
At 15 docs	0.3054
At 20 docs	0.2802
At 30 docs	0.2442
At 100 docs	0.1602
At 200 docs	0.1198
At 500 docs	0.0749
At 1000 docs	0.0472
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2458



Summary Statistics	
Run Number	sdmix3-category B, automatic
Number of Topics	43
Total number of documents over all topics	
Retrieved:	43000
Relevant:	2689
Rel_Ret:	1649

Recall Level Precision Averages	
Recall	Precision
0.00	0.3757
0.10	0.2521
0.20	0.2047
0.30	0.1657
0.40	0.1409
0.50	0.1162
0.60	0.0814
0.70	0.0733
0.80	0.0568
0.90	0.0284
1.00	0.0236
Average precision over all relevant docs	
non-interpolated	0.1261

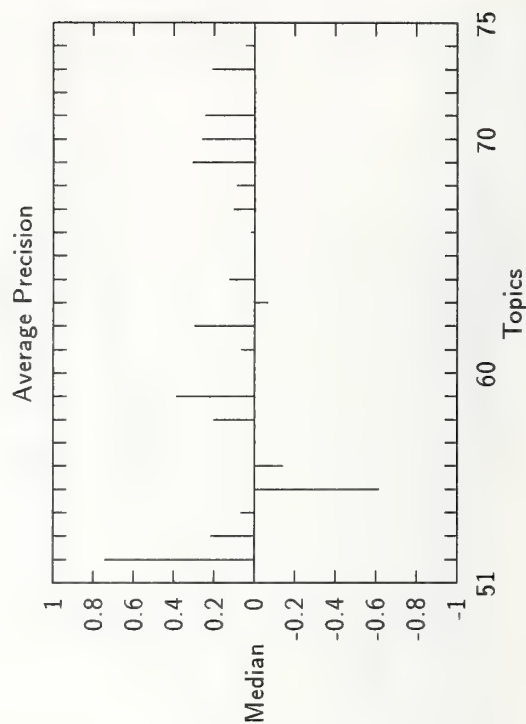
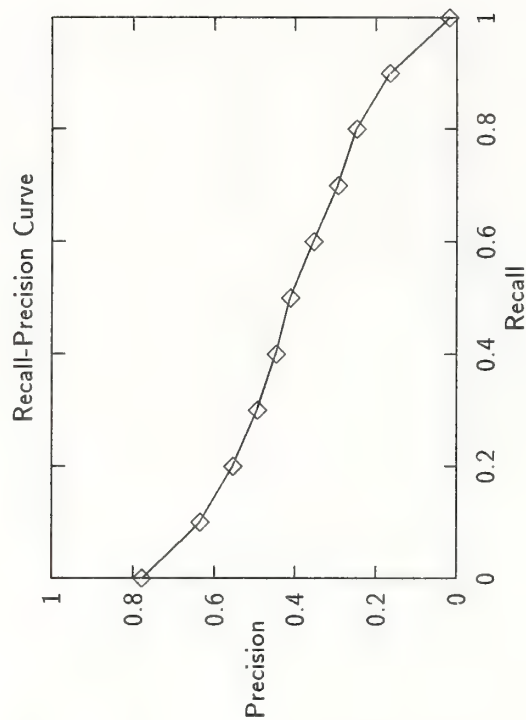
Document Level Averages	
At 5 docs	0.2047
At 10 docs	0.2047
At 15 docs	0.1969
At 20 docs	0.1837
At 30 docs	0.1674
At 100 docs	0.1300
At 200 docs	0.0960
At 500 docs	0.0572
At 1000 docs	0.0383
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1521



Summary Statistics	
Run Number	Cor5SPIs-automatic, short topic
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel_ret:	2103

Recall Level Precision Averages	
Recall	Precision
0.00	0.7798
0.10	0.6367
0.20	0.5556
0.30	0.4952
0.40	0.4488
0.50	0.4128
0.60	0.3560
0.70	0.2952
0.80	0.2500
0.90	0.1666
1.00	0.0198
Average precision over all relevant docs	
non-interpolated	0.3949

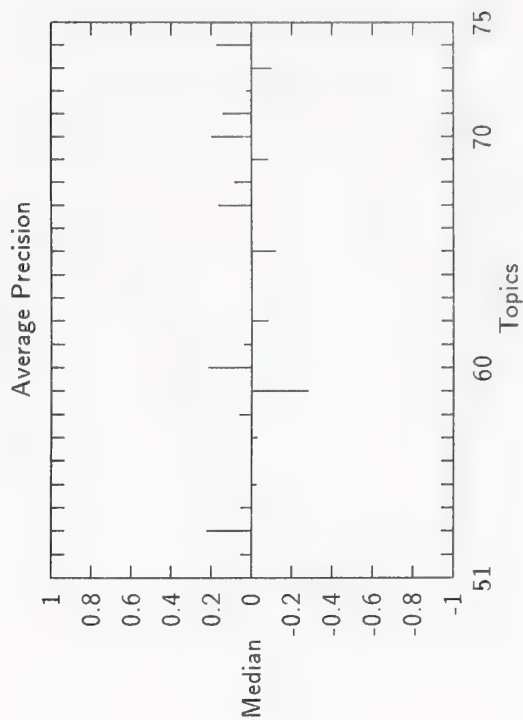
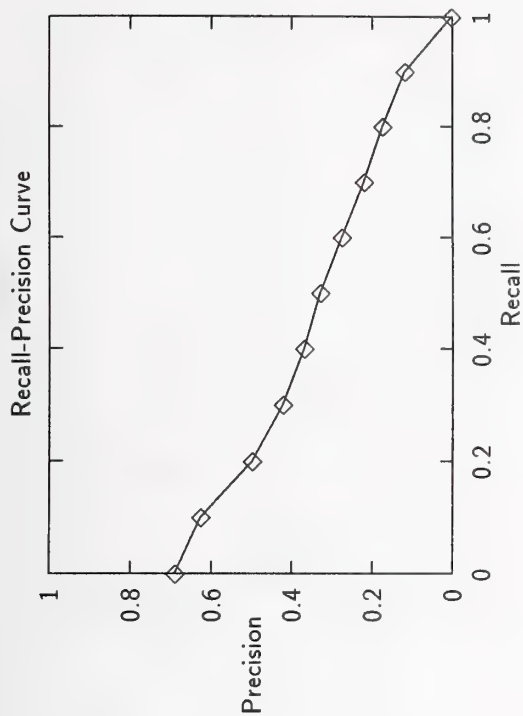
Document Level Averages	
	Precision
At 5 docs	0.6160
At 10 docs	0.5480
At 15 docs	0.4987
At 20 docs	0.4740
At 30 docs	0.4333
At 100 docs	0.3448
At 200 docs	0.2624
At 500 docs	0.1469
At 1000 docs	0.0841
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3890



Summary Statistics	
Run Number	DCU966-automatic, short topic
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel_ret:	2109

Recall Level Precision Averages		
Recall	Precision	
0.00	0.6905	
0.10	0.6268	
0.20	0.4985	
0.30	0.4227	
0.40	0.3700	
0.50	0.3299	
0.60	0.2773	
0.70	0.2217	
0.80	0.1744	
0.90	0.1192	
1.00	0.0026	
Average precision over all relevant docs		
non-interpolated	0.3298	

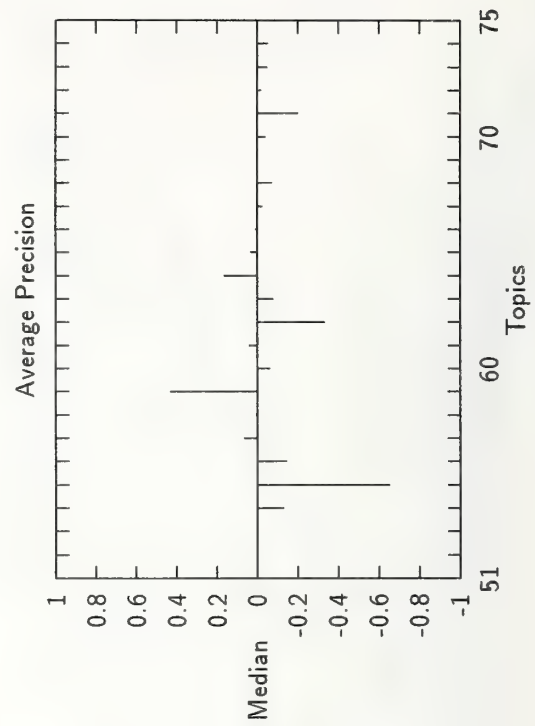
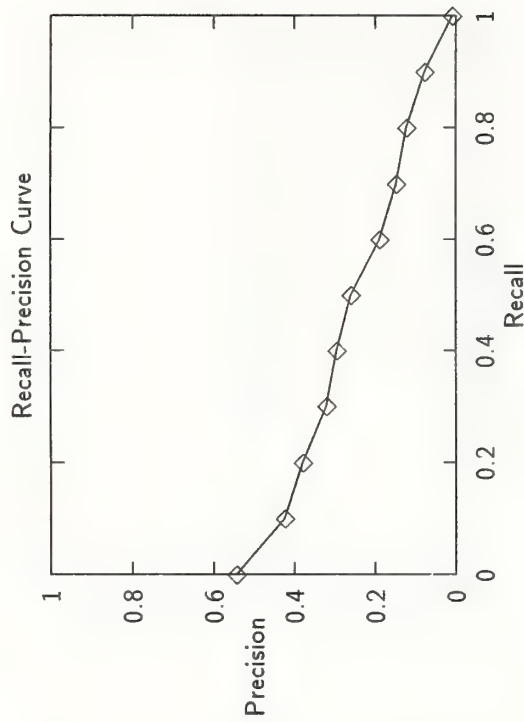
Document Level Averages	
At 5 docs	0.4800
At 10 docs	0.4960
At 15 docs	0.5013
At 20 docs	0.4840
At 30 docs	0.4507
At 100 docs	0.3352
At 200 docs	0.2524
At 500 docs	0.1446
At 1000 docs	0.0844
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3553



Summary Statistics	
Run Number	gmu96sp1-automatic, short topic
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel_ret:	1689

Recall Level Precision Averages	
Recall	Precision
0.00	0.5441
0.10	0.4237
0.20	0.3798
0.30	0.3218
0.40	0.2965
0.50	0.2601
0.60	0.1884
0.70	0.1482
0.80	0.1222
0.90	0.0786
1.00	0.0102
Average precision over all relevant docs	
non-interpolated	0.2403

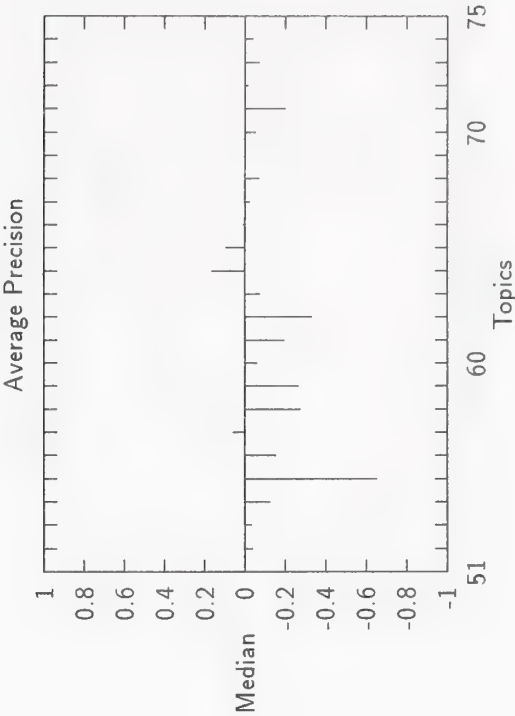
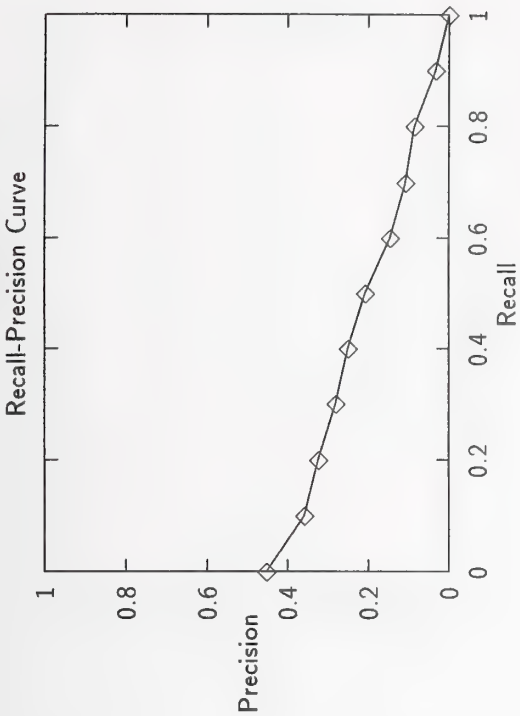
Document Level Averages	
	Precision
At 5 docs	0.3760
At 10 docs	0.3520
At 15 docs	0.3173
At 20 docs	0.3020
At 30 docs	0.2773
At 100 docs	0.2160
At 200 docs	0.1716
At 500 docs	0.1089
At 1000 docs	0.0676
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2482



Summary Statistics		
Run Number	gmu96sp2-automatic, short topic	25
Total number of documents over all topics		
Retrieved:	25000	
Relevant:	2505	
Rel_ret:	1606	

Recall Level Precision Averages	
Recall	Precision
0.00	0.4538
0.10	0.3600
0.20	0.3246
0.30	0.2809
0.40	0.2504
0.50	0.2087
0.60	0.1458
0.70	0.1076
0.80	0.0865
0.90	0.0336
1.00	0.0003
Average precision over all relevant docs	
non-interpolated	0.1900

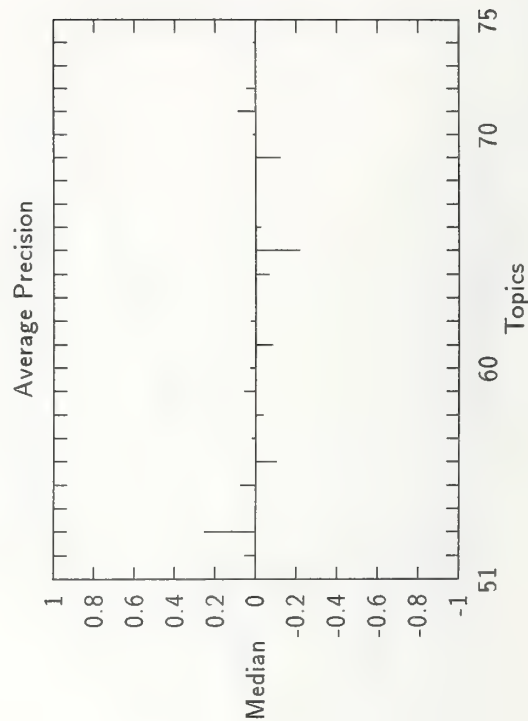
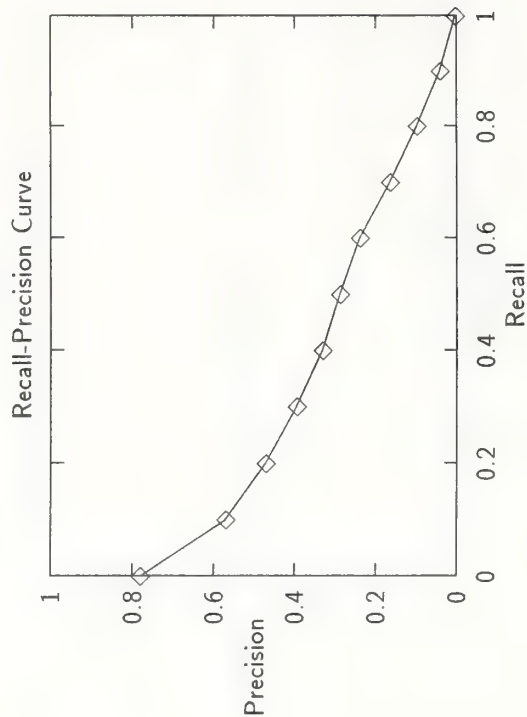
Document Level Averages	
At 5 docs	0.2240
At 10 docs	0.2320
At 15 docs	0.2293
At 20 docs	0.2420
At 30 docs	0.2400
At 100 docs	0.1924
At 200 docs	0.1570
At 500 docs	0.1006
At 1000 docs	0.0642
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2149



Summary Statistics	
Run Number	nmsuc1-automatic, short topic
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel_ret:	1892

Recall Level Precision Averages	
Recall	Precision
0.00	0.7812
0.10	0.5712
0.20	0.4729
0.30	0.3962
0.40	0.3315
0.50	0.2890
0.60	0.2402
0.70	0.1653
0.80	0.0981
0.90	0.0408
1.00	0.0026
Average precision over all relevant docs	
non-interpolated	0.2895

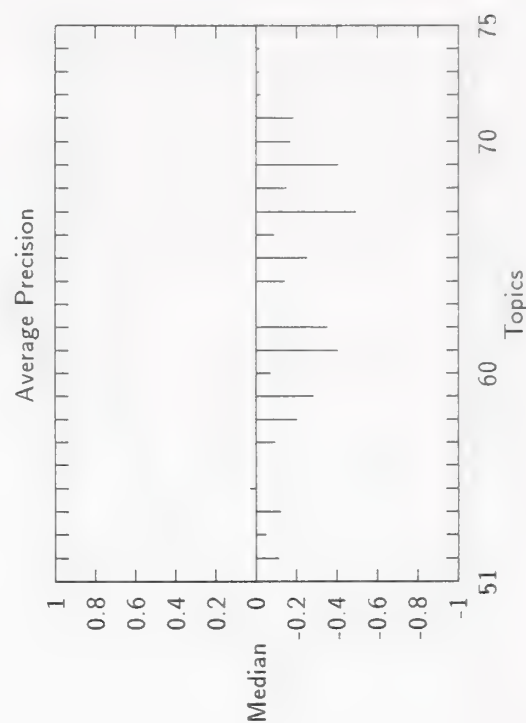
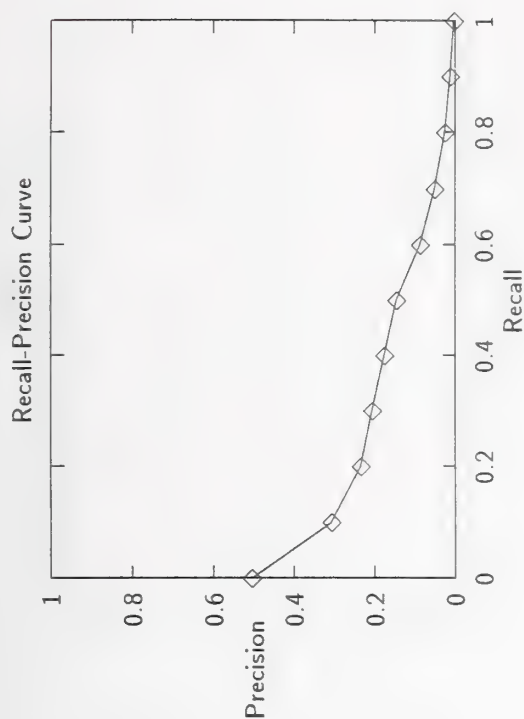
Document Level Averages	
	Precision
At 5 docs	0.5520
At 10 docs	0.4880
At 15 docs	0.4533
At 20 docs	0.4240
At 30 docs	0.3947
At 100 docs	0.3024
At 200 docs	0.2284
At 500 docs	0.1279
At 1000 docs	0.0757
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3397



Summary Statistics	
Run Number	nmsuc2-automatic, short topic
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel.ret:	1305

Recall Level Precision Averages	
Recall	Precision
0.00	0.5058
0.10	0.3078
0.20	0.2346
0.30	0.2063
0.40	0.1768
0.50	0.1458
0.60	0.0882
0.70	0.0511
0.80	0.0251
0.90	0.0132
1.00	0.0032
Average precision over all relevant docs	
non-interpolated	0.1422

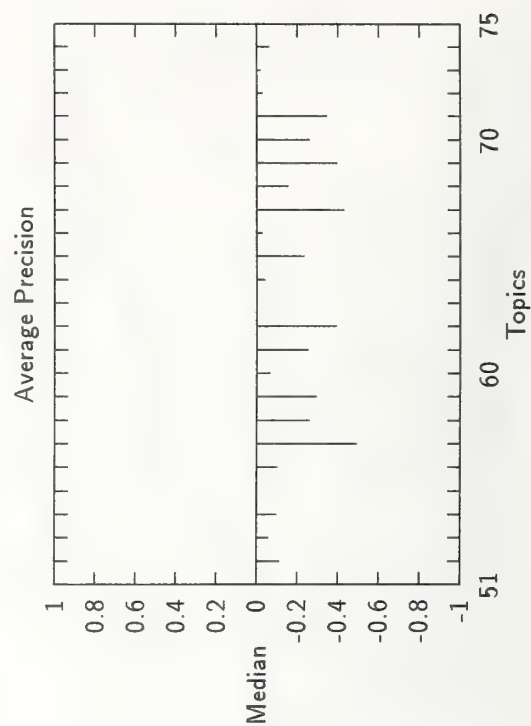
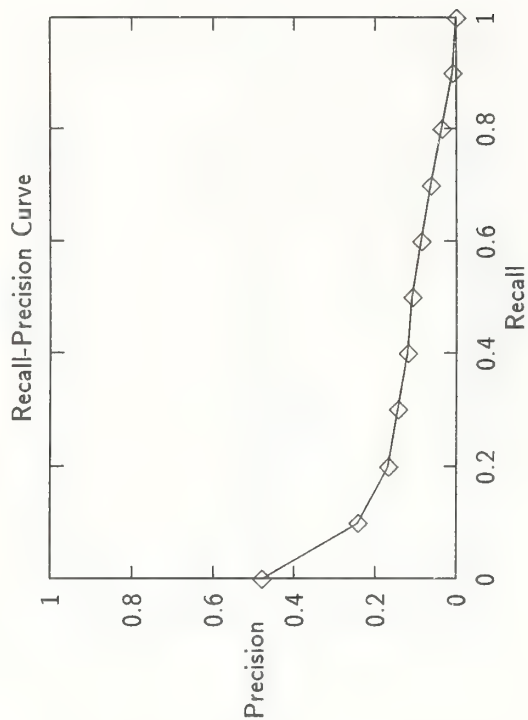
Document Level Averages	
	Precision
At 5 docs	0.2800
At 10 docs	0.2560
At 15 docs	0.2400
At 20 docs	0.2240
At 30 docs	0.2187
At 100 docs	0.1880
At 200 docs	0.1414
At 500 docs	0.0836
At 1000 docs	0.0522
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1834



Summary Statistics	
Run Number	nmsuc3-automatic, short topic
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel.ret:	1210

Recall Level Precision Averages	
Recall	Precision
0.00	0.4804
0.10	0.2429
0.20	0.1687
0.30	0.1439
0.40	0.1216
0.50	0.1098
0.60	0.0873
0.70	0.0629
0.80	0.0363
0.90	0.0099
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.1153

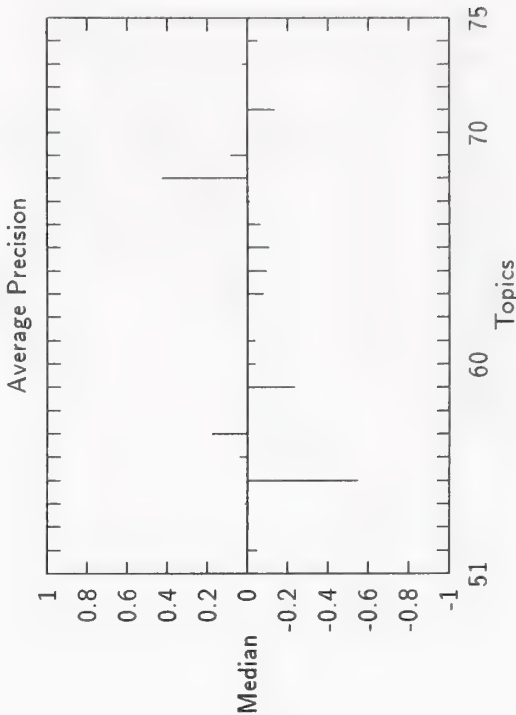
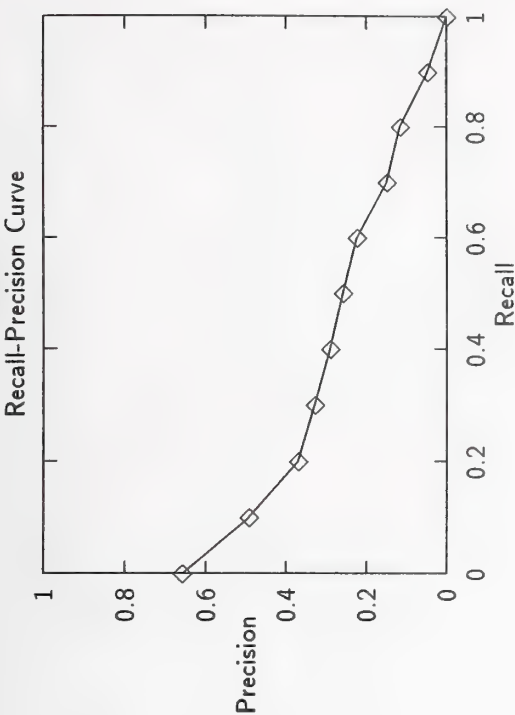
Document Level Averages	
	Precision
At 5 docs	0.2800
At 10 docs	0.2320
At 15 docs	0.2240
At 20 docs	0.2100
At 30 docs	0.1840
At 100 docs	0.1496
At 200 docs	0.1176
At 500 docs	0.0737
At 1000 docs	0.0484
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1420



Summary Statistics	
Run Number	BrklySP5-automatic, short topic
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel_ret:	1632

Recall Level Precision Averages	
Recall	Precision
0.00	0.6583
0.10	0.4935
0.20	0.3701
0.30	0.3298
0.40	0.2902
0.50	0.2594
0.60	0.2248
0.70	0.1502
0.80	0.1166
0.90	0.0469
1.00	0.0005
Average precision over all relevant docs	
non-interpolated	0.2526

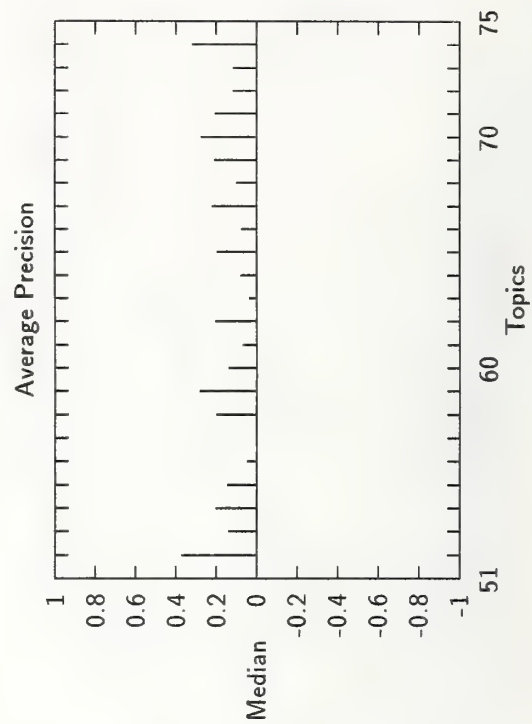
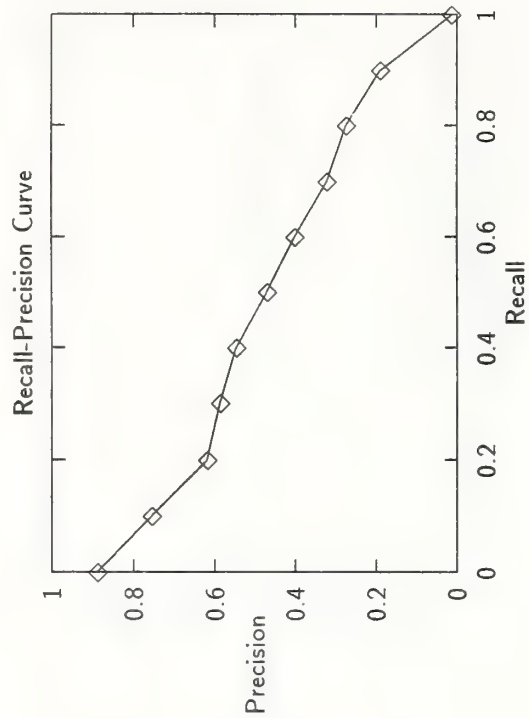
Document Level Averages	
	Precision
At 5 docs	0.4640
At 10 docs	0.4200
At 15 docs	0.3920
At 20 docs	0.3600
At 30 docs	0.3373
At 100 docs	0.2424
At 200 docs	0.1754
At 500 docs	0.1043
At 1000 docs	0.0653
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2766



Summary Statistics	
Run Number	SIN300-automatic, short topic
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel_ret:	2320

Recall Level Precision Averages	
Recall	Precision
0.00	0.8885
0.10	0.7545
0.20	0.6182
0.30	0.5873
0.40	0.5466
0.50	0.4711
0.60	0.4015
0.70	0.3234
0.80	0.2748
0.90	0.1909
1.00	0.0141
Average precision over all relevant docs	
non-interpolated	0.4551

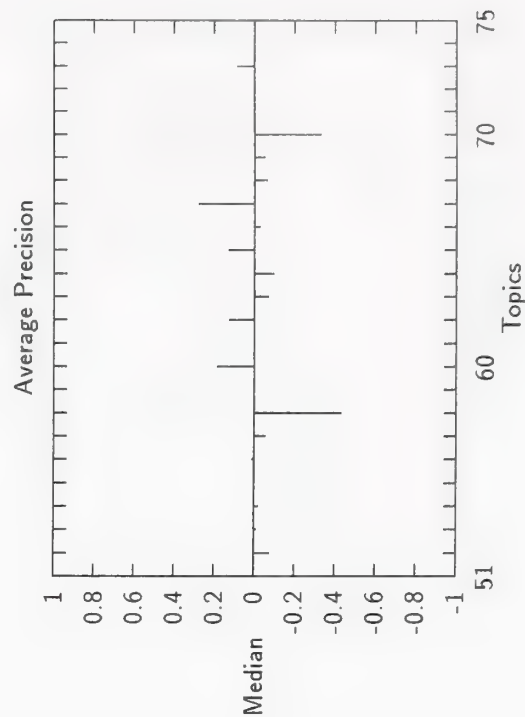
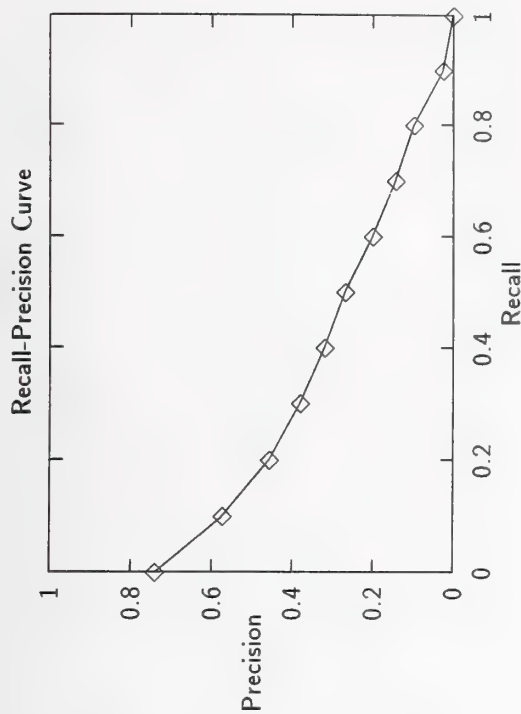
Document Level Averages	
	Precision
At 5 docs	0.6720
At 10 docs	0.6200
At 15 docs	0.5787
At 20 docs	0.5400
At 30 docs	0.5253
At 100 docs	0.4044
At 200 docs	0.3066
At 500 docs	0.1683
At 1000 docs	0.0928
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4474



Summary Statistics	
Run Number	xerox-spD-automatic, short topic
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel_ret:	1475

Recall Level Precision Averages	
Recall	Precision
0.00	0.7409
0.10	0.5744
0.20	0.4581
0.30	0.3811
0.40	0.3212
0.50	0.2703
0.60	0.2019
0.70	0.1453
0.80	0.0992
0.90	0.0259
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.2745

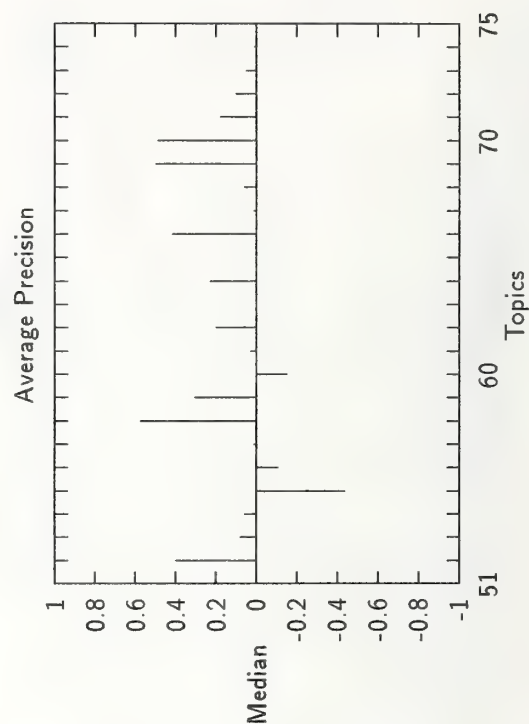
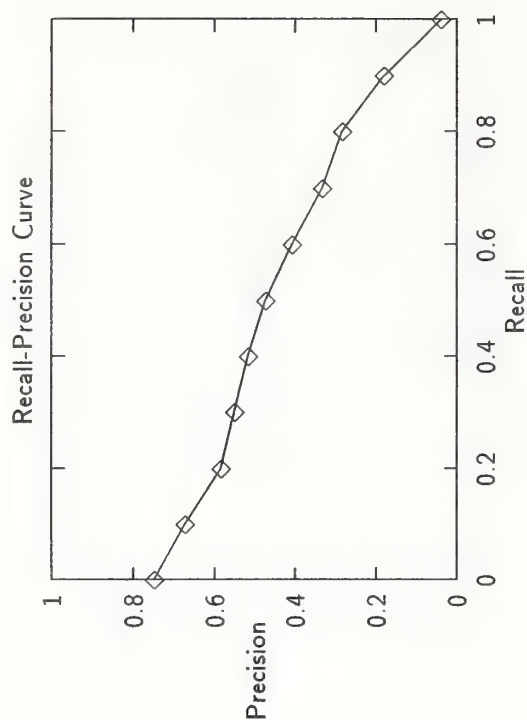
Document Level Averages	
	Precision
At 5 docs	0.5280
At 10 docs	0.4480
At 15 docs	0.4267
At 20 docs	0.4120
At 30 docs	0.3747
At 100 docs	0.2760
At 200 docs	0.2010
At 500 docs	0.1054
At 1000 docs	0.0590
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3258



Summary Statistics	
Run Number	Cor5SP21-automatic, long topic
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel_ret:	2222

Recall Level Precision Averages	
Recall	Precision
0.00	0.7482
0.10	0.6729
0.20	0.5840
0.30	0.5505
0.40	0.5172
0.50	0.4739
0.60	0.4081
0.70	0.3338
0.80	0.2842
0.90	0.1820
1.00	0.0393
Average precision over all relevant docs	
non-interpolated	0.4323

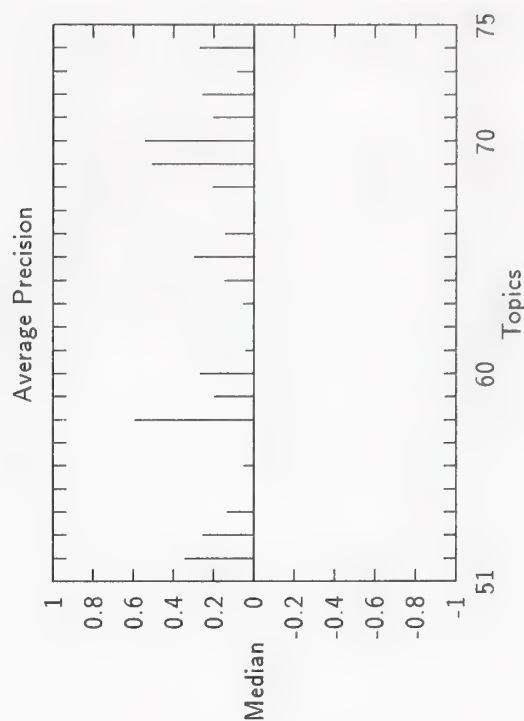
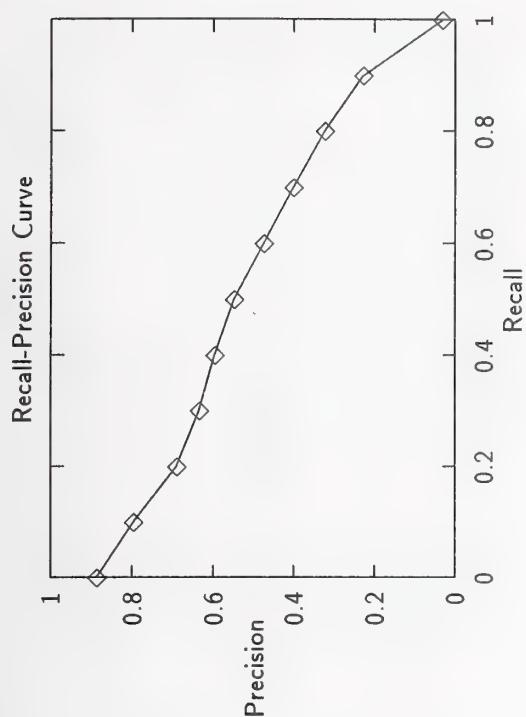
Document Level Averages	
	Precision
At 5 docs	0.6080
At 10 docs	0.5640
At 15 docs	0.5307
At 20 docs	0.5020
At 30 docs	0.4787
At 100 docs	0.3796
At 200 docs	0.2988
At 500 docs	0.1606
At 1000 docs	0.0889
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4468



Summary Statistics	
Run Number	SIN301-automatic, long topic
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel_ret:	2377

Recall Level Precision Averages	
Recall	Precision
0.00	0.8864
0.10	0.7962
0.20	0.6899
0.30	0.6343
0.40	0.5969
0.50	0.5482
0.60	0.4755
0.70	0.4030
0.80	0.3241
0.90	0.2287
1.00	0.0312
Average precision over all relevant docs	
non-interpolated	0.5058

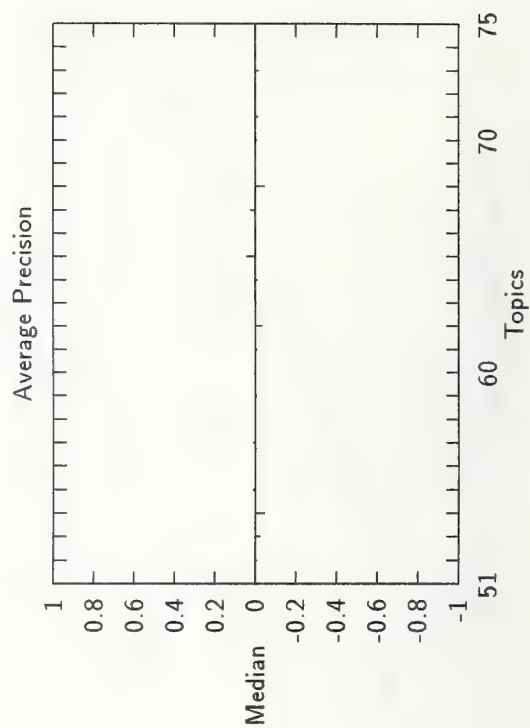
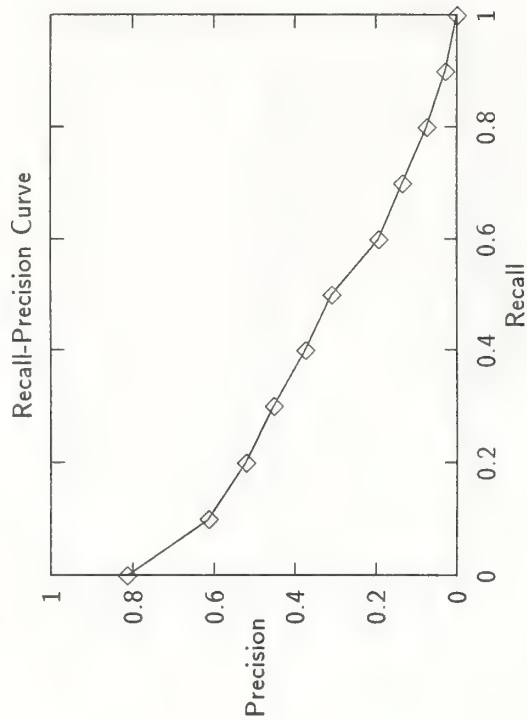
Document Level Averages	
	Precision
At 5 docs	0.6800
At 10 docs	0.6680
At 15 docs	0.6347
At 20 docs	0.6080
At 30 docs	0.5893
At 100 docs	0.4532
At 200 docs	0.3382
At 500 docs	0.1769
At 1000 docs	0.0951
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.5106



Summary Statistics	
Run Number	xerox-spP-automatic, long topic
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel_ret:	1496

Recall Level Precision Averages	
Recall	Precision
0.00	0.8139
0.10	0.6144
0.20	0.5209
0.30	0.4538
0.40	0.3748
0.50	0.3101
0.60	0.1944
0.70	0.1354
0.80	0.0751
0.90	0.0282
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.3005

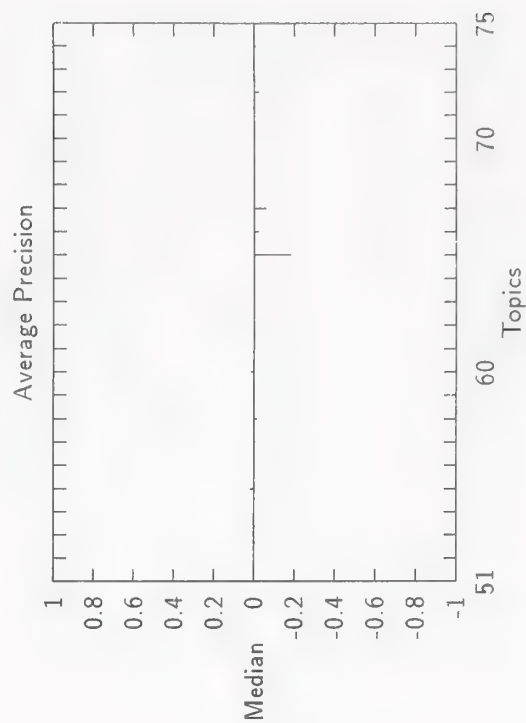
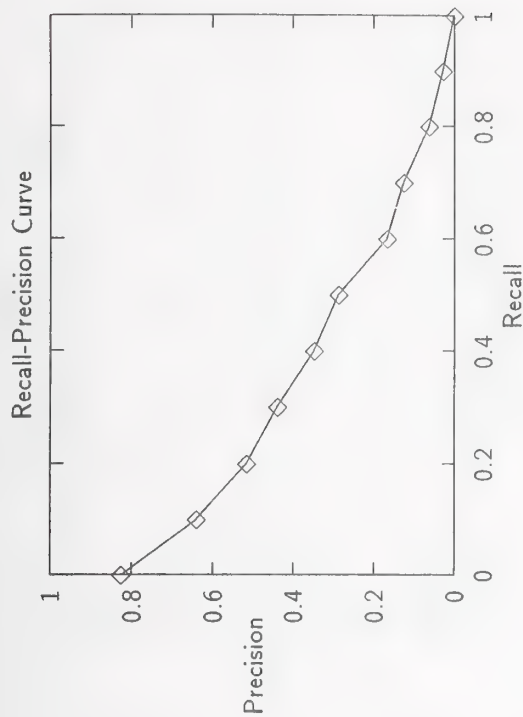
Document Level Averages	
At 5 docs	0.5600
At 10 docs	0.5280
At 15 docs	0.4960
At 20 docs	0.4600
At 30 docs	0.4213
At 100 docs	0.3244
At 200 docs	0.2268
At 500 docs	0.1095
At 1000 docs	0.0598
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3500



Summary Statistics		
Run Number	xerox-spS-automatic, long topic	25
Number of Topics		
Total number of documents over all topics		
Retrieved:		25000
Relevant:		2505
Rel-ret:		1497

Recall Level Precision Averages	
Recall	Precision
0.00	0.8264
0.10	0.6407
0.20	0.5159
0.30	0.4409
0.40	0.3503
0.50	0.2886
0.60	0.1685
0.70	0.1270
0.80	0.0640
0.90	0.0281
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.2946

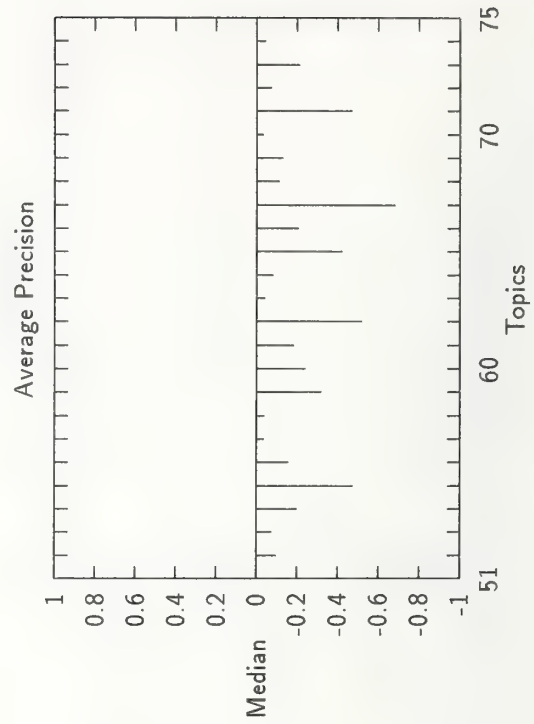
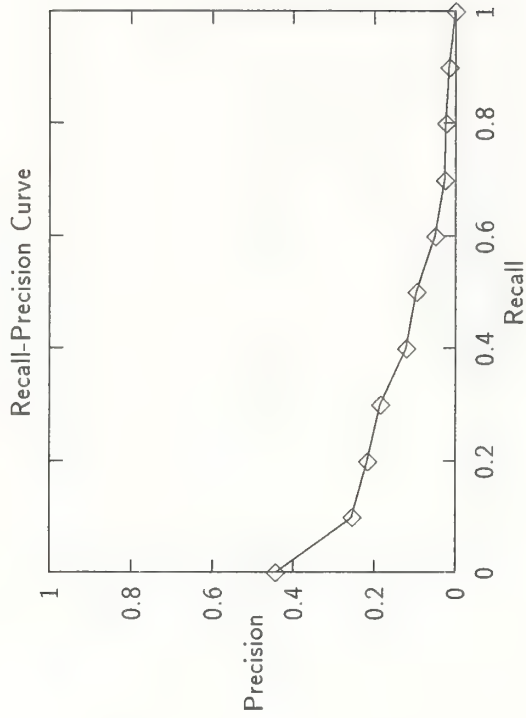
Document Level Averages	
	Precision
At 5 docs	0.5920
At 10 docs	0.5320
At 15 docs	0.4800
At 20 docs	0.4600
At 30 docs	0.4293
At 100 docs	0.3212
At 200 docs	0.2272
At 500 docs	0.1111
At 1000 docs	0.0599
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3376



Summary Statistics		
Run Number	xerox-sp T-automatic, long topic	25
Total number of documents over all topics		
Retrieved:	25000	
Relevant:	2505	
Rel_ret:	1115	

Recall Level Precision Averages	
Recall	Precision
0.00	0.4454
0.10	0.2553
0.20	0.2185
0.30	0.1863
0.40	0.1230
0.50	0.0972
0.60	0.0503
0.70	0.0276
0.80	0.0238
0.90	0.0157
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.1154

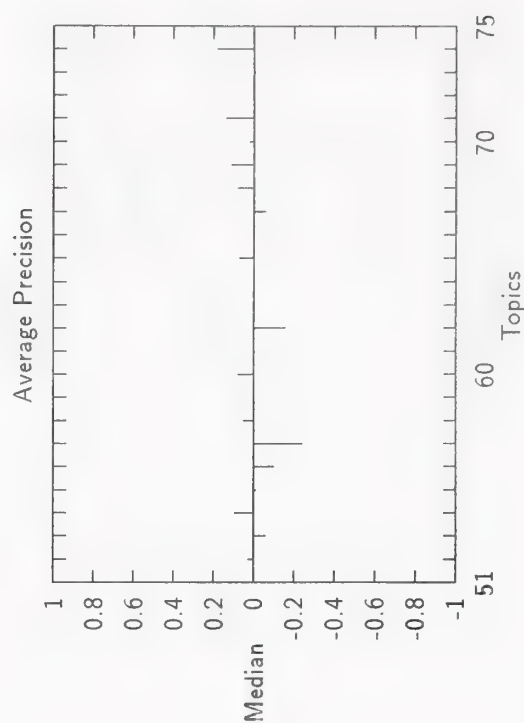
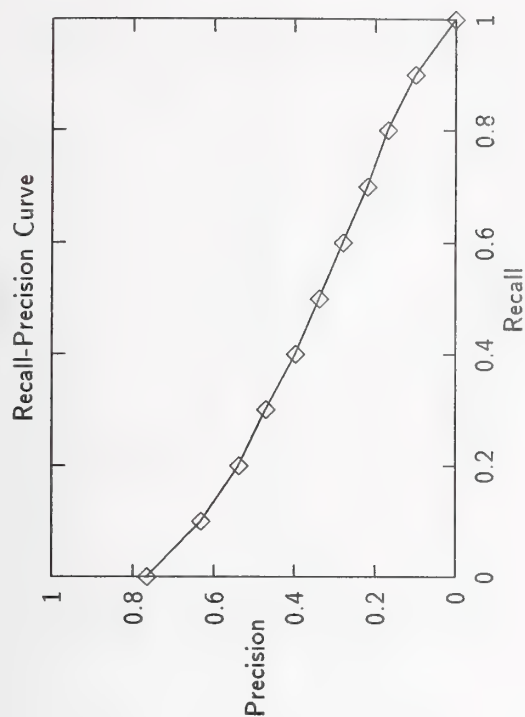
Document Level Averages	
At 5 docs	0.2800
At 10 docs	0.2480
At 15 docs	0.2427
At 20 docs	0.2340
At 30 docs	0.2213
At 100 docs	0.1744
At 200 docs	0.1314
At 500 docs	0.0766
At 1000 docs	0.0446
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1547



Summary Statistics	
Run Number	DCU965-manual
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel.ret:	2157

Recall Level Precision Averages	
Recall	Precision
0.00	0.7661
0.10	0.6335
0.20	0.5383
0.30	0.4719
0.40	0.3985
0.50	0.3404
0.60	0.2819
0.70	0.2213
0.80	0.1697
0.90	0.1020
1.00	0.0021
Average precision over all relevant docs	
non-interpolated	0.3482

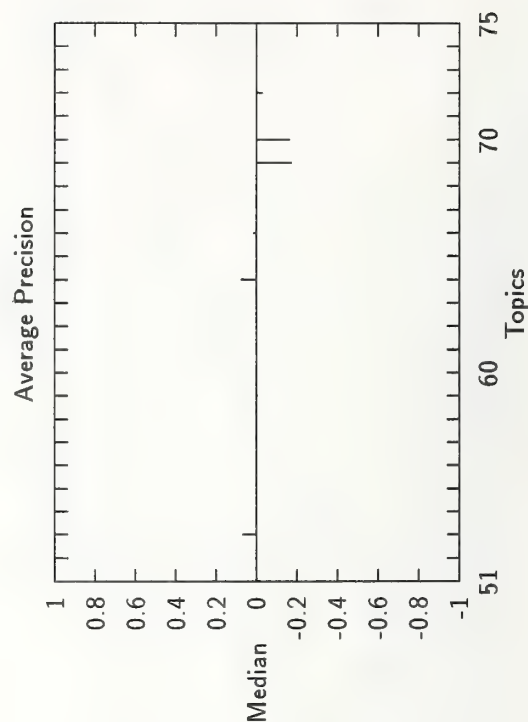
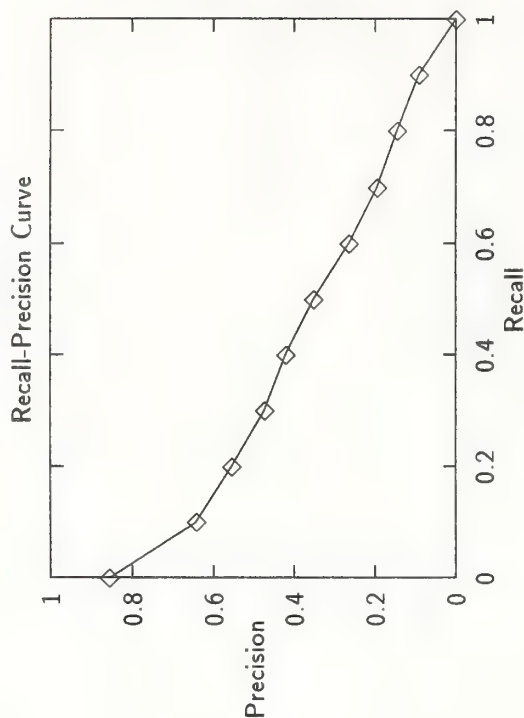
Document Level Averages	
	Precision
At 5 docs	0.5840
At 10 docs	0.5760
At 15 docs	0.5360
At 20 docs	0.5100
At 30 docs	0.4813
At 100 docs	0.3676
At 200 docs	0.2734
At 500 docs	0.1488
At 1000 docs	0.0863
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3729



Summary Statistics	
Run Number	DCU967-manual
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel_ret:	2037

Recall Level Precision Averages	
Recall	Precision
0.00	0.8565
0.10	0.6428
0.20	0.5547
0.30	0.4733
0.40	0.4213
0.50	0.3533
0.60	0.2660
0.70	0.1954
0.80	0.1468
0.90	0.0927
1.00	0.0004
Average precision over all relevant docs	
non-interpolated	0.3482

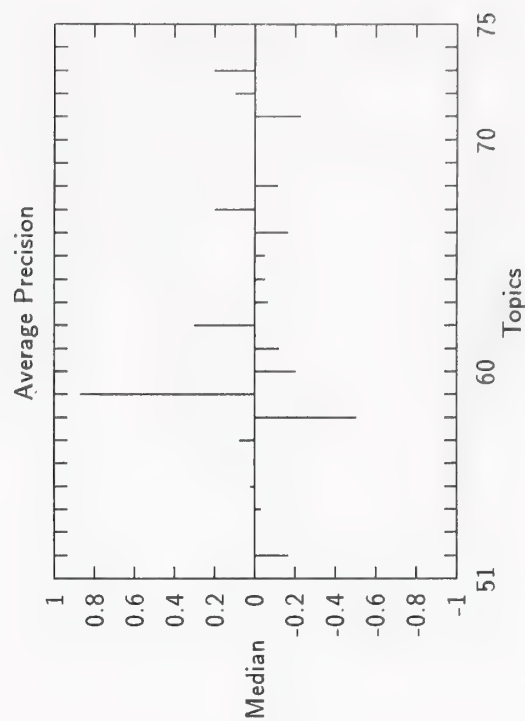
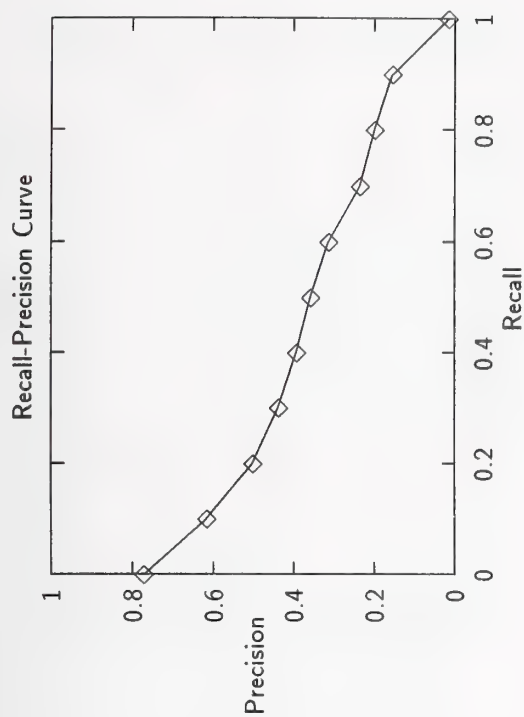
Document Level Averages	
	Precision
At 5 docs	0.6080
At 10 docs	0.5760
At 15 docs	0.5067
At 20 docs	0.4820
At 30 docs	0.4427
At 100 docs	0.3612
At 200 docs	0.2626
At 500 docs	0.1431
At 1000 docs	0.0815
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3733



Summary Statistics	
Run Number	BrklySP6-manual
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2505
Rel_ret:	2103

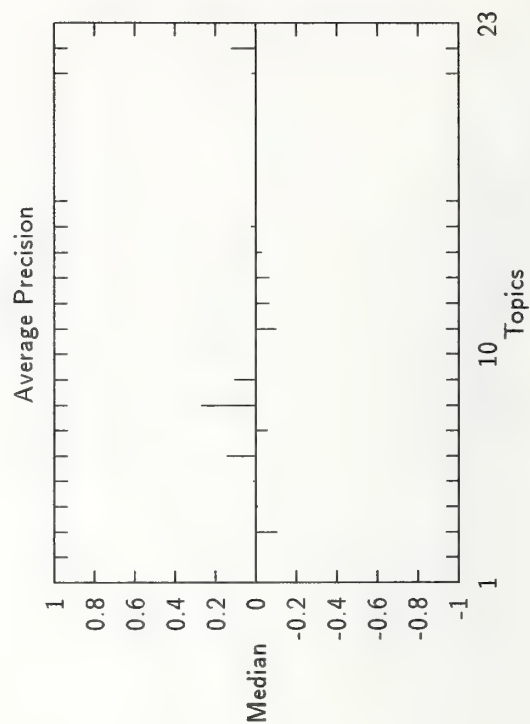
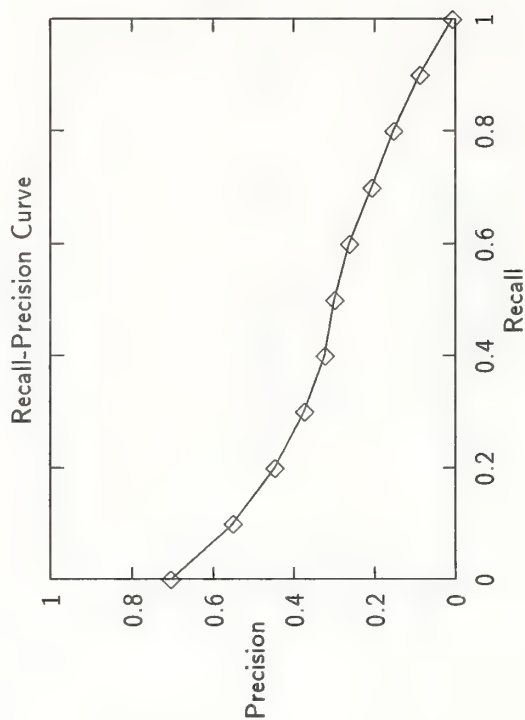
Recall Level Precision Averages	
Recall	Precision
0.00	0.7726
0.10	0.6171
0.20	0.5017
0.30	0.4387
0.40	0.3946
0.50	0.3596
0.60	0.3151
0.70	0.2363
0.80	0.2002
0.90	0.1561
1.00	0.0160
Average precision over all relevant docs	
non-interpolated	0.3488

Document Level Averages	
	Precision
At 5 docs	0.5120
At 10 docs	0.5040
At 15 docs	0.4693
At 20 docs	0.4560
At 30 docs	0.4320
At 100 docs	0.3040
At 200 docs	0.2242
At 500 docs	0.1346
At 1000 docs	0.0841
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3822



Summary Statistics	
Run Number	city96c1-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
Rel.ret:	1203

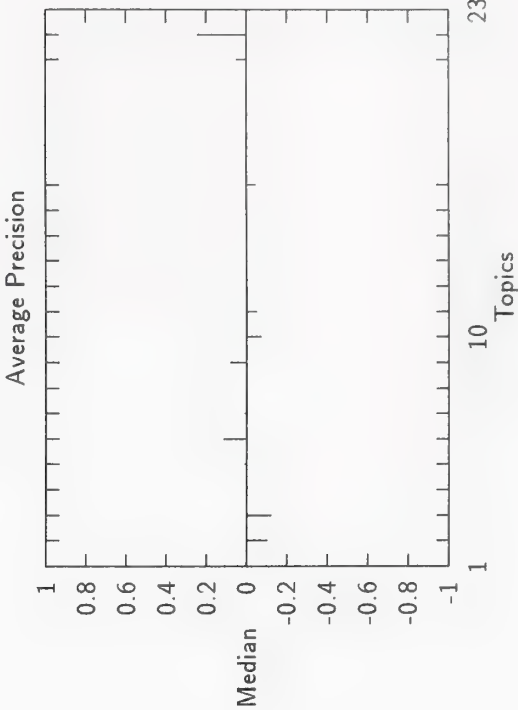
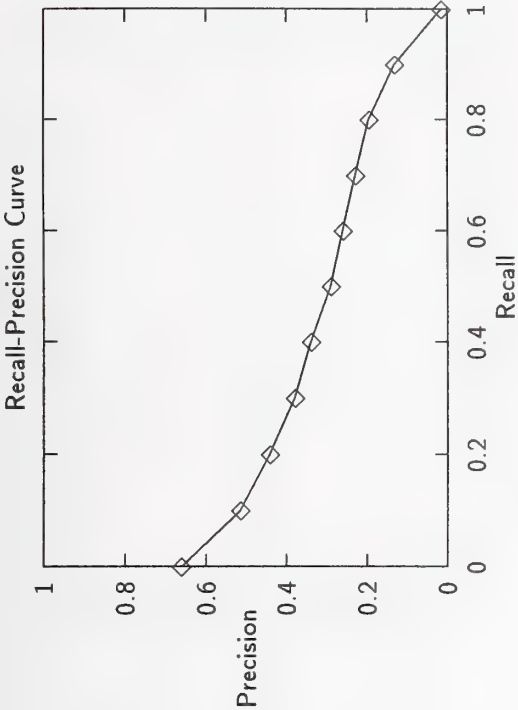
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.7063	At 5 docs	0.4842
0.10	0.5513	At 10 docs	0.4632
0.20	0.4477	At 15 docs	0.4175
0.30	0.3753	At 20 docs	0.3895
0.40	0.3239	At 30 docs	0.3561
0.50	0.3007	At 100 docs	0.2589
0.60	0.2639	At 200 docs	0.1971
0.70	0.2078	At 500 docs	0.1099
0.80	0.1546	At 1000 docs	0.0633
0.90	0.0897	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0087		
Average precision over all relevant docs		Exact	0.3050
non-interpolated	0.2943		



Summary Statistics	
Run Number	city96c2-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
Rel_ret:	1258

Recall Level Precision Averages	
Recall	Precision
0.00	0.6618
0.10	0.5146
0.20	0.4412
0.30	0.3807
0.40	0.3400
0.50	0.2916
0.60	0.2614
0.70	0.2289
0.80	0.1953
0.90	0.1326
1.00	0.0153
Average precision over all relevant docs	
non-interpolated	0.2933

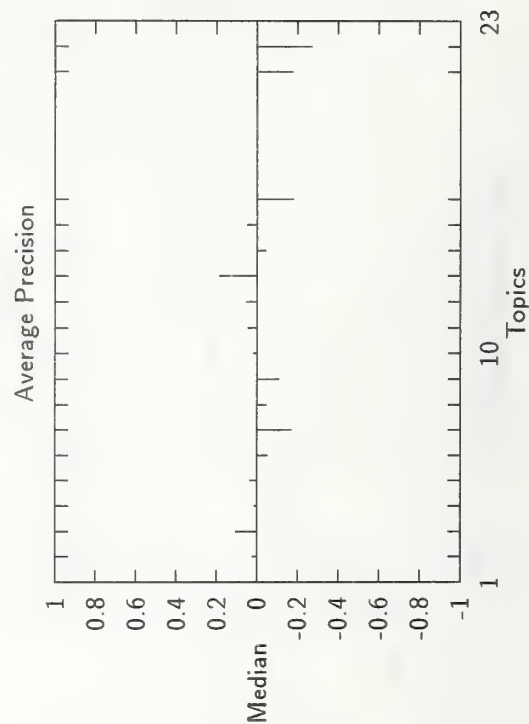
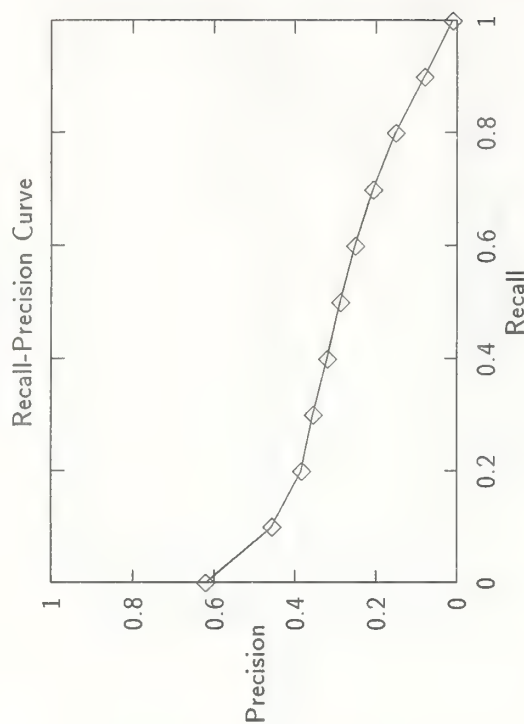
Document Level Averages	
	Precision
At 5 docs	0.4316
At 10 docs	0.3842
At 15 docs	0.3895
At 20 docs	0.3737
At 30 docs	0.3456
At 100 docs	0.2600
At 200 docs	0.2037
At 500 docs	0.1174
At 1000 docs	0.0662
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3055



Summary Statistics	
Run Number	CLCHNA-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
Rel_ret:	1182

Recall Level Precision Averages	
Recall	Precision
0.00	0.6223
0.10	0.4576
0.20	0.3854
0.30	0.3561
0.40	0.3206
0.50	0.2888
0.60	0.2527
0.70	0.2086
0.80	0.1533
0.90	0.0807
1.00	0.0109
Average precision over all relevant docs	
non-interpolated	0.2677

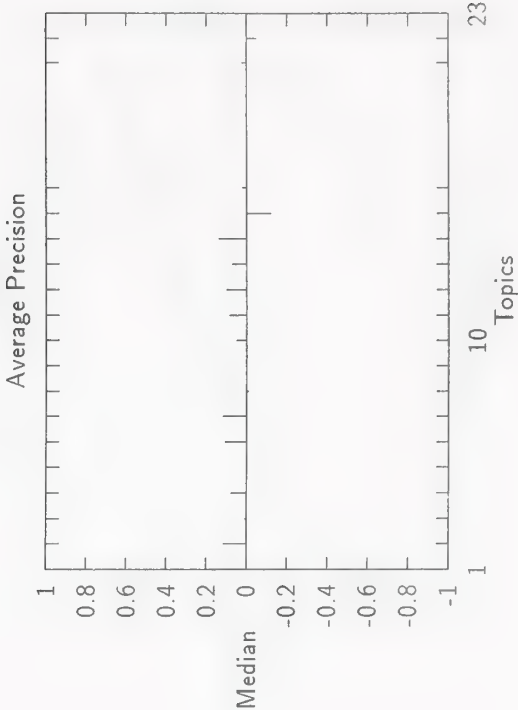
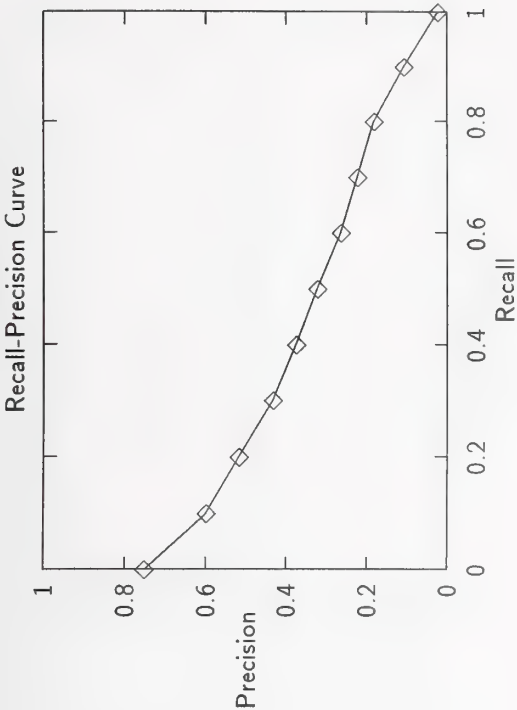
Document Level Averages	
	Precision
At 5 docs	0.3789
At 10 docs	0.3947
At 15 docs	0.3930
At 20 docs	0.3895
At 30 docs	0.3561
At 100 docs	0.2653
At 200 docs	0.2063
At 500 docs	0.1123
At 1000 docs	0.0622
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2998



Summary Statistics	
Run Number	Cor5C1vt-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
Rel_ret:	1286

Recall Level Precision Averages	
Recall	Precision
0.00	0.7516
0.10	0.5991
0.20	0.5186
0.30	0.4341
0.40	0.3769
0.50	0.3245
0.60	0.2652
0.70	0.2238
0.80	0.1825
0.90	0.1086
1.00	0.0243
Average precision over all relevant docs	
non-interpolated	0.3266

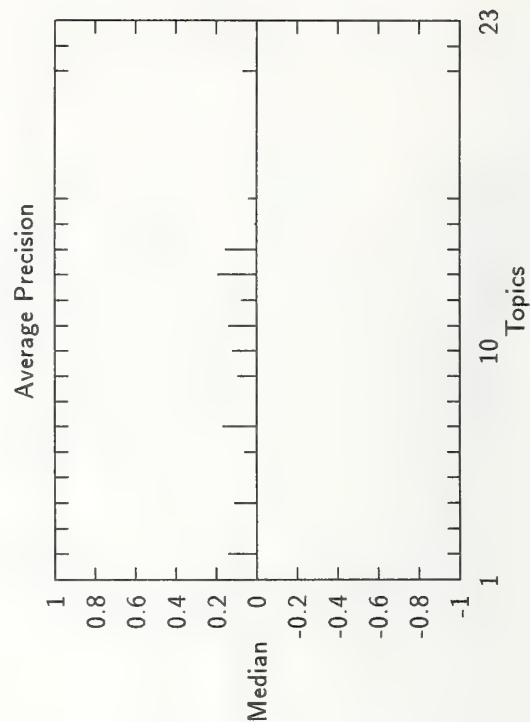
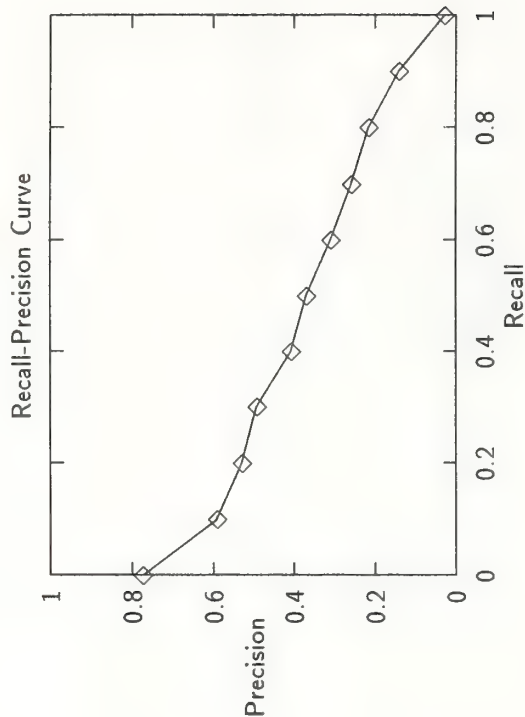
Document Level Averages	
	Precision
At 5 docs	0.4842
At 10 docs	0.5105
At 15 docs	0.4632
At 20 docs	0.4368
At 30 docs	0.4175
At 100 docs	0.3026
At 200 docs	0.2279
At 500 docs	0.1222
At 1000 docs	0.0677
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3598



Summary Statistics	
Run Number	Cor5C2ex-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
RelRet:	1343

Recall Level Precision Averages	
Recall	Precision
0.00	0.7744
0.10	0.5910
0.20	0.5302
0.30	0.4922
0.40	0.4079
0.50	0.3712
0.60	0.3100
0.70	0.2594
0.80	0.2165
0.90	0.1421
1.00	0.0284
Average precision over all relevant docs	
non-interpolated	0.3598

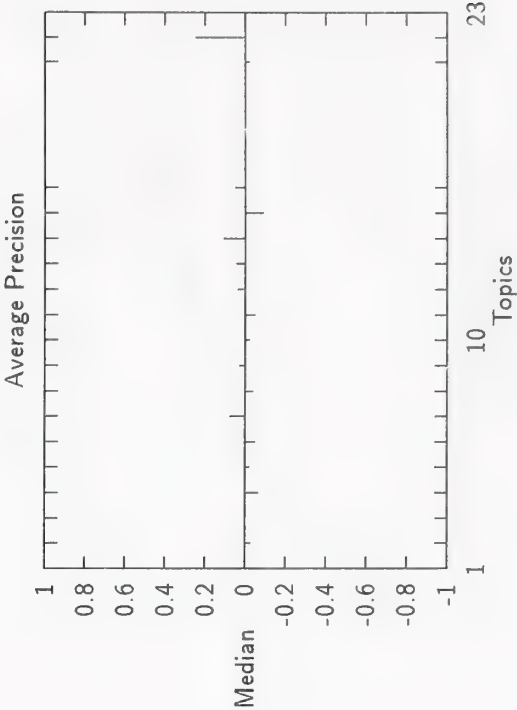
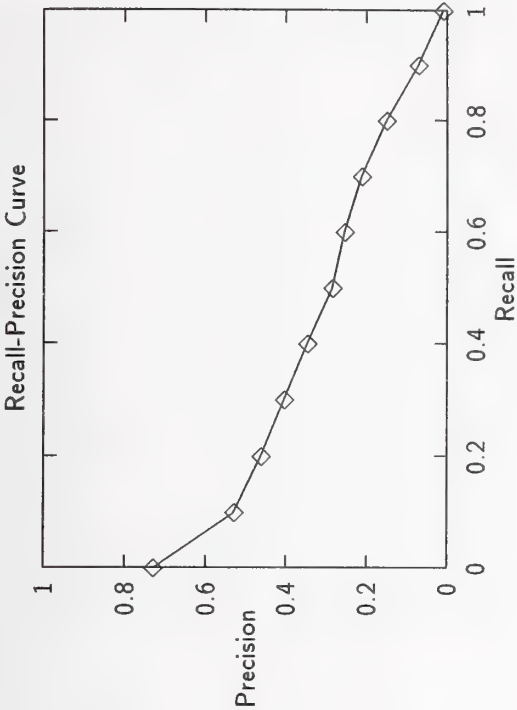
Document Level Averages	
	Precision
At 5 docs	0.5579
At 10 docs	0.4947
At 15 docs	0.4702
At 20 docs	0.4553
At 30 docs	0.4263
At 100 docs	0.3084
At 200 docs	0.2403
At 500 docs	0.1300
At 1000 docs	0.0707
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3829



Summary Statistics	
Run Number	gmu96ca1-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
RelRet:	1202

Recall Level Precision Averages	
Recall	Precision
0.00	0.7304
0.10	0.5294
0.20	0.4621
0.30	0.4057
0.40	0.3479
0.50	0.2858
0.60	0.2566
0.70	0.2128
0.80	0.1515
0.90	0.0712
1.00	0.0088
Average precision over all relevant docs	
non-interpolated	0.2955

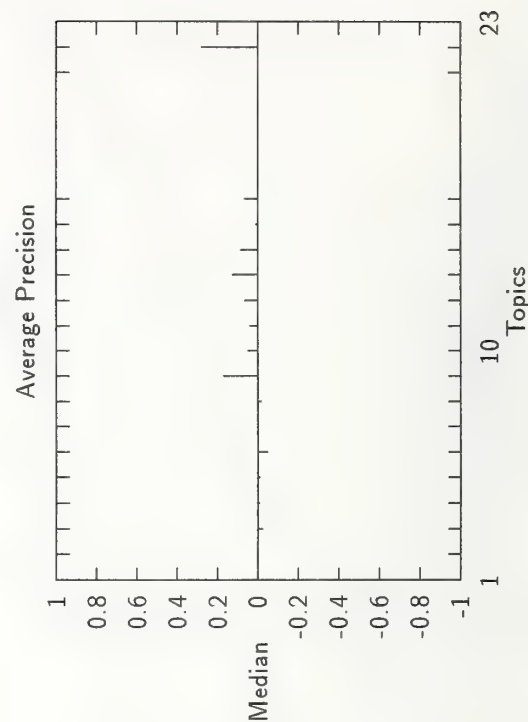
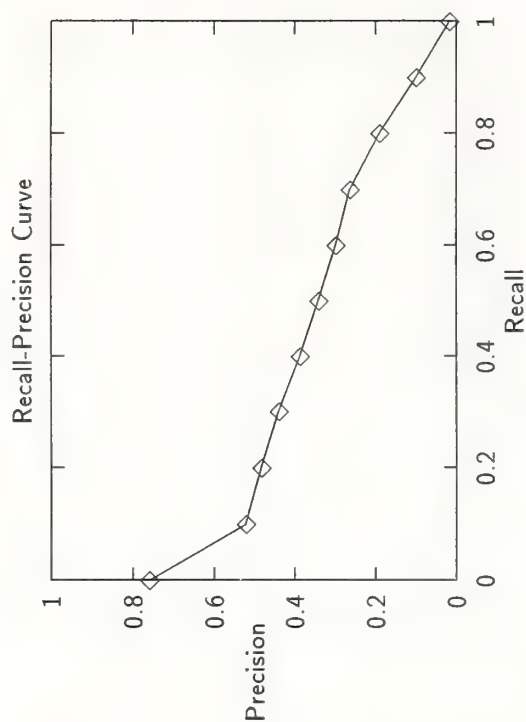
Document Level Averages	
At 5 docs	0.4737
At 10 docs	0.4263
At 15 docs	0.4140
At 20 docs	0.4000
At 30 docs	0.3667
At 100 docs	0.2595
At 200 docs	0.1947
At 500 docs	0.1082
At 1000 docs	0.0633
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3296



Summary Statistics	
Run Number	gmu96ca2-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
Rel_ret:	1250

Recall Level Precision Averages		
Recall	Precision	
0.00	0.7600	
0.10	0.5210	
0.20	0.4838	
0.30	0.4407	
0.40	0.3875	
0.50	0.3419	
0.60	0.2992	
0.70	0.2632	
0.80	0.1911	
0.90	0.1000	
1.00	0.0171	
Average precision over all relevant docs		
non-interpolated	0.3274	

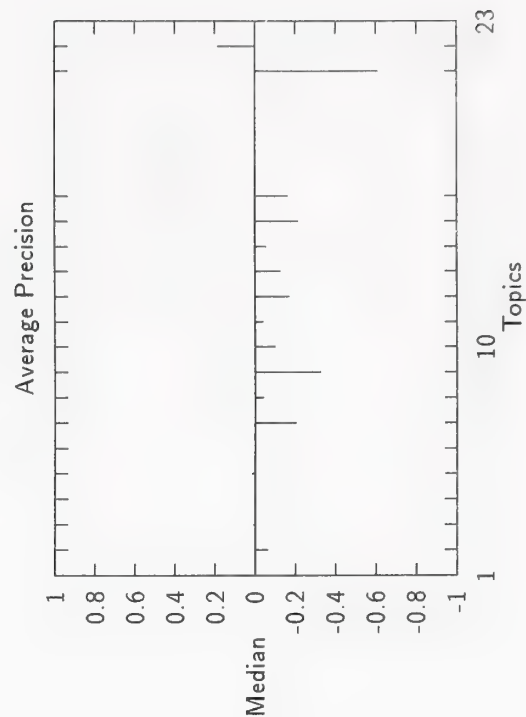
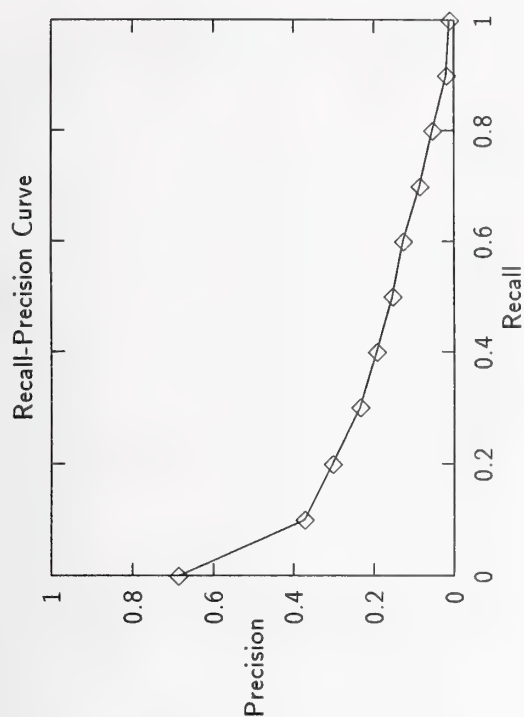
Document Level Averages	
	Precision
At 5 docs	0.4947
At 10 docs	0.4579
At 15 docs	0.4281
At 20 docs	0.4342
At 30 docs	0.4175
At 100 docs	0.2916
At 200 docs	0.2147
At 500 docs	0.1169
At 1000 docs	0.0658
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3571



Summary Statistics	
Run Number	itcn1-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	17501
Relevant:	1399
Rel_ret:	899

Recall Level Precision Averages	
Recall	Precision
0.00	0.6861
0.10	0.3744
0.20	0.3039
0.30	0.2355
0.40	0.1945
0.50	0.1543
0.60	0.1270
0.70	0.0863
0.80	0.0541
0.90	0.0185
1.00	0.0119
Average precision over all relevant docs	
non-interpolated	0.1731

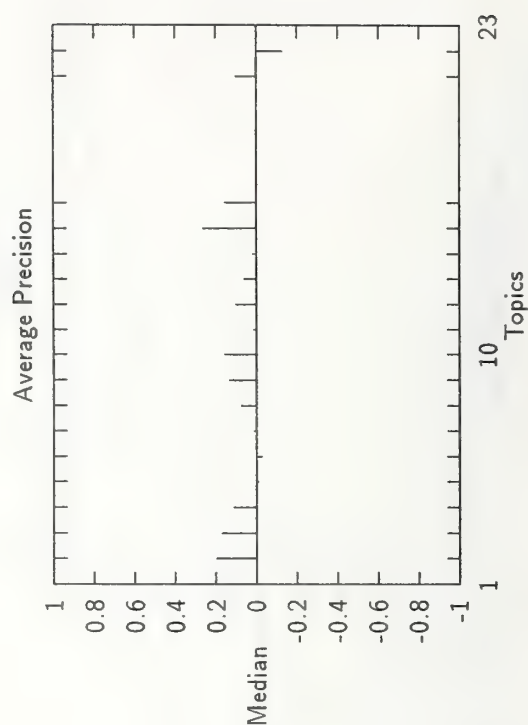
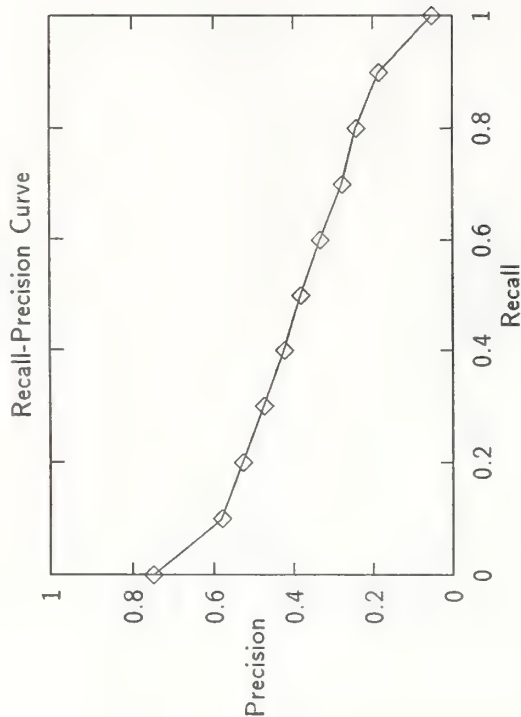
Document Level Averages	
	Precision
At 5 docs	0.3263
At 10 docs	0.3105
At 15 docs	0.2912
At 20 docs	0.2737
At 30 docs	0.2316
At 100 docs	0.1758
At 200 docs	0.1282
At 500 docs	0.0763
At 1000 docs	0.0473
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2289



Summary Statistics	
Run Number	pircsCw-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
Rel.Ret:	1297

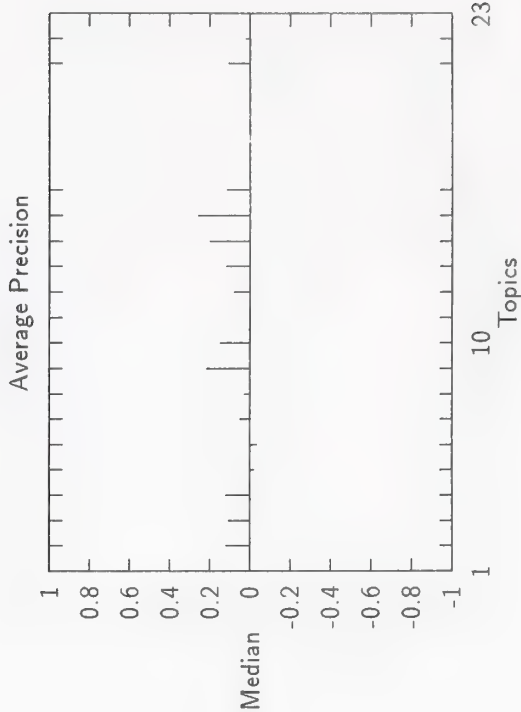
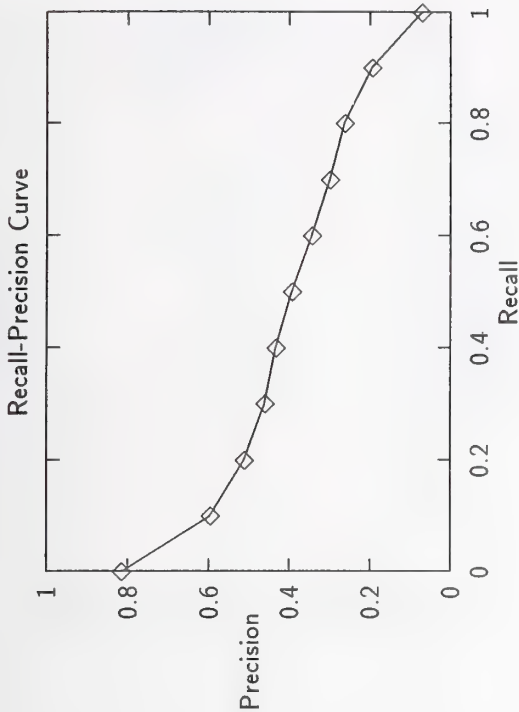
Recall Level Precision Averages	
Recall	Precision
0.00	0.7497
0.10	0.5809
0.20	0.5272
0.30	0.4747
0.40	0.4243
0.50	0.3831
0.60	0.3339
0.70	0.2784
0.80	0.2425
0.90	0.1859
1.00	0.0525
Average precision over all relevant docs	
non-interpolated	0.3751

Document Level Averages	
	Precision
At 5 docs	0.5263
At 10 docs	0.4947
At 15 docs	0.4877
At 20 docs	0.4895
At 30 docs	0.4474
At 100 docs	0.3211
At 200 docs	0.2366
At 500 docs	0.1246
At 1000 docs	0.0683
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3870



Summary Statistics	
Run Number	pircsCwc-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
Rel_ret:	1313

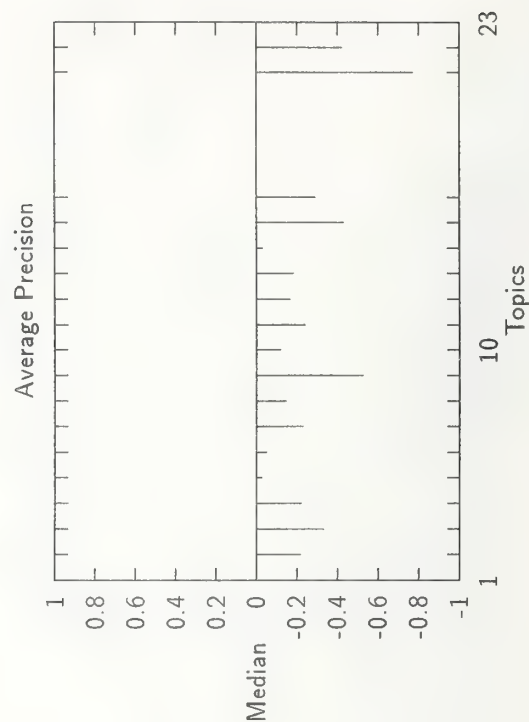
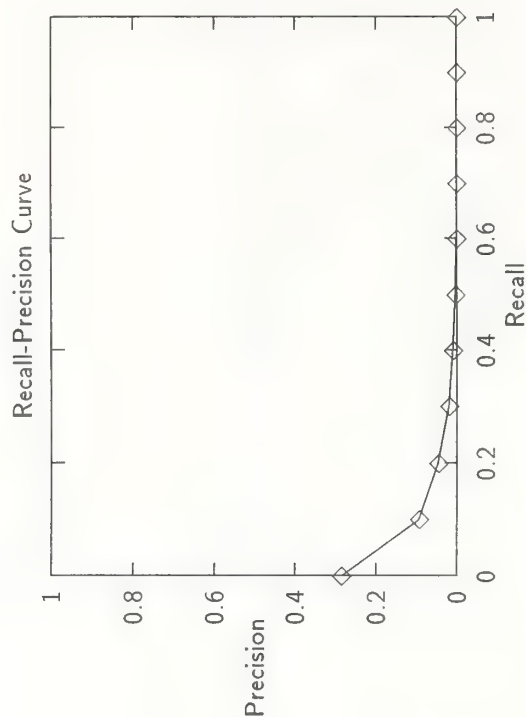
Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.8150	At 5 docs	0.5053
0.10	0.5958	At 10 docs	0.4895
0.20	0.5116	At 15 docs	0.4737
0.30	0.4614	At 20 docs	0.4421
0.40	0.4335	At 30 docs	0.4211
0.50	0.3940	At 100 docs	0.3253
0.60	0.3453	At 200 docs	0.2400
0.70	0.2998	At 500 docs	0.1279
0.80	0.2637	At 1000 docs	0.0691
0.90	0.1957	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.0714		
Average precision over all relevant docs			
non-interpolated	0.3789	Exact	0.3823



Summary Statistics	
Run Number	mds004-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	18999
Relevant:	1399
Rel_ret:	360

Recall Level Precision Averages	
Recall	Precision
0.00	0.2854
0.10	0.0937
0.20	0.0454
0.30	0.0187
0.40	0.0091
0.50	0.0039
0.60	0.0000
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0268

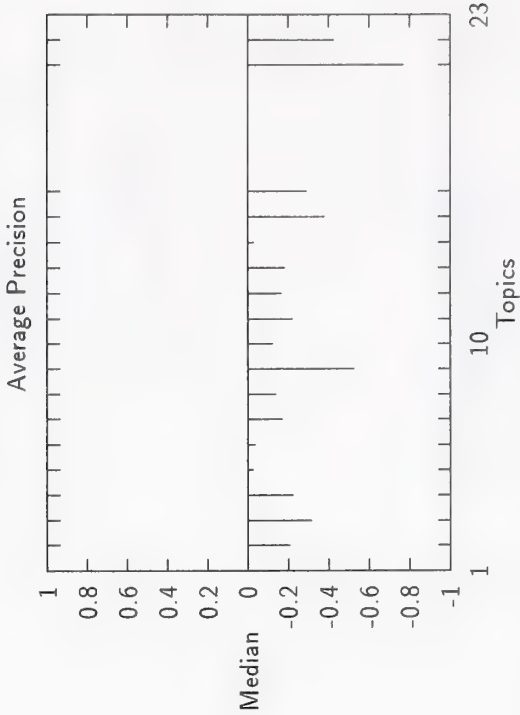
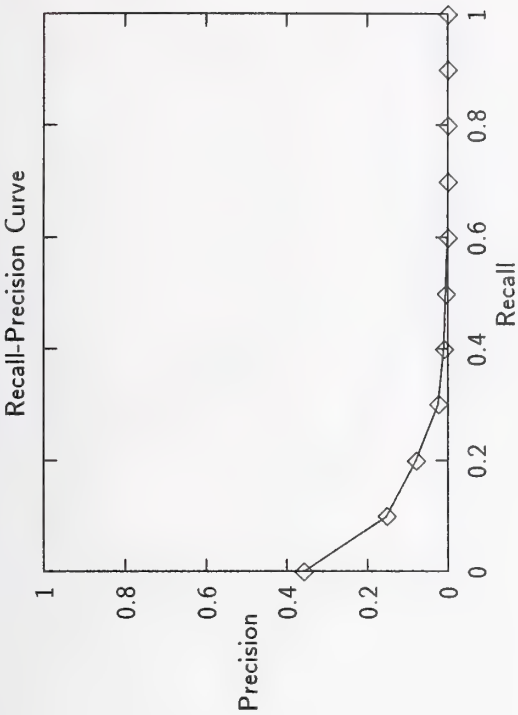
Document Level Averages	
	Precision
At 5 docs	0.0947
At 10 docs	0.1158
At 15 docs	0.1018
At 20 docs	0.0921
At 30 docs	0.0930
At 100 docs	0.0705
At 200 docs	0.0476
At 500 docs	0.0292
At 1000 docs	0.0189
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0867



Summary Statistics	
Run Number	mds005-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	18999
Relevant:	1399
Rel_ret:	360

Recall Level Precision Averages	
Recall	Precision
0.00	0.3597
0.10	0.1534
0.20	0.0790
0.30	0.0241
0.40	0.0100
0.50	0.0043
0.60	0.0000
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0371

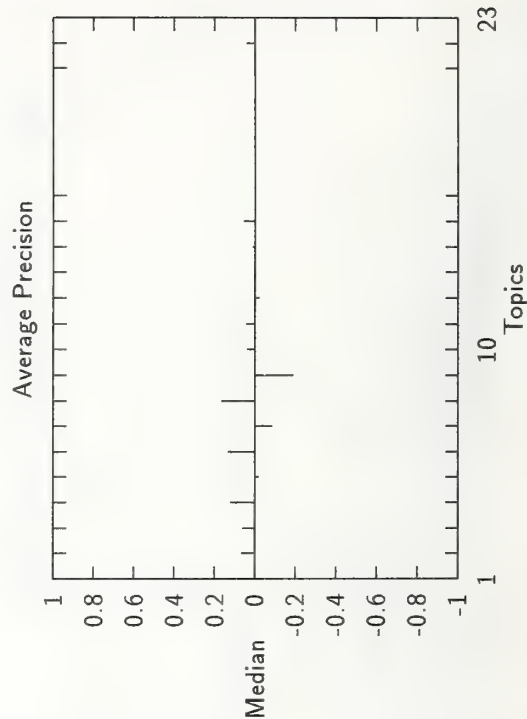
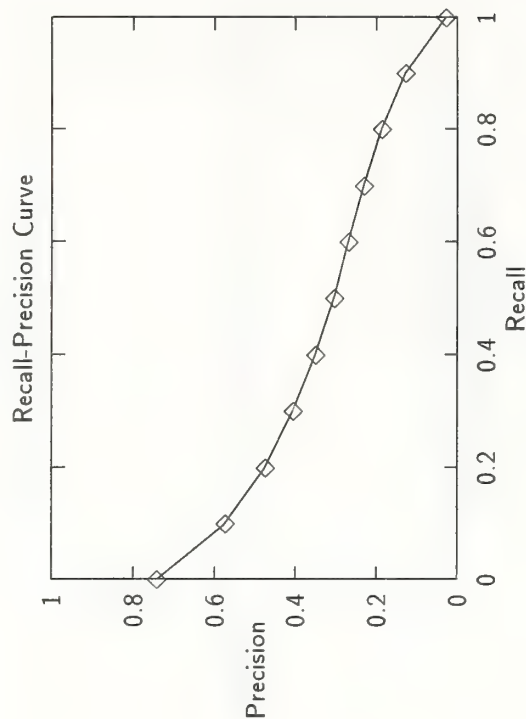
Document Level Averages	
At 5 docs	0.1474
At 10 docs	0.1211
At 15 docs	0.1298
At 20 docs	0.1237
At 30 docs	0.1123
At 100 docs	0.0811
At 200 docs	0.0558
At 500 docs	0.0338
At 1000 docs	0.0189
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1045



Summary Statistics	
Run Number	BrklyCH1-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
Rel_ret:	1246

Recall Level Precision Averages	
Recall	Precision
0.00	0.7429
0.10	0.5745
0.20	0.4758
0.30	0.4068
0.40	0.3513
0.50	0.3051
0.60	0.2709
0.70	0.2326
0.80	0.1874
0.90	0.1293
1.00	0.0280
Average precision over all relevant docs	
non-interpolated	0.3192

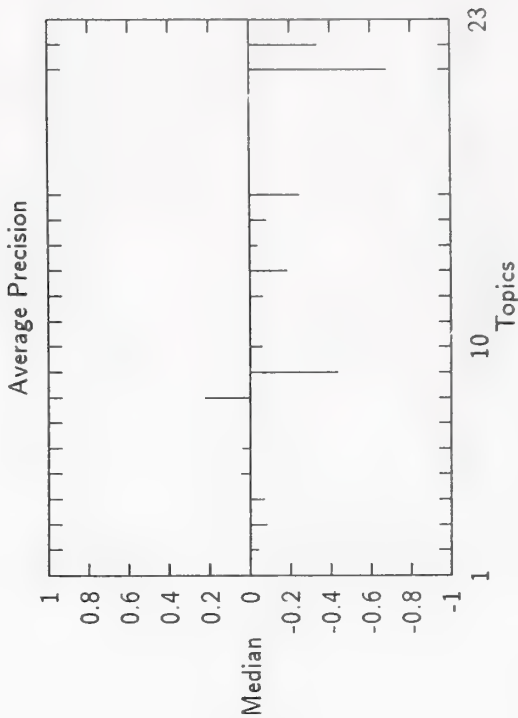
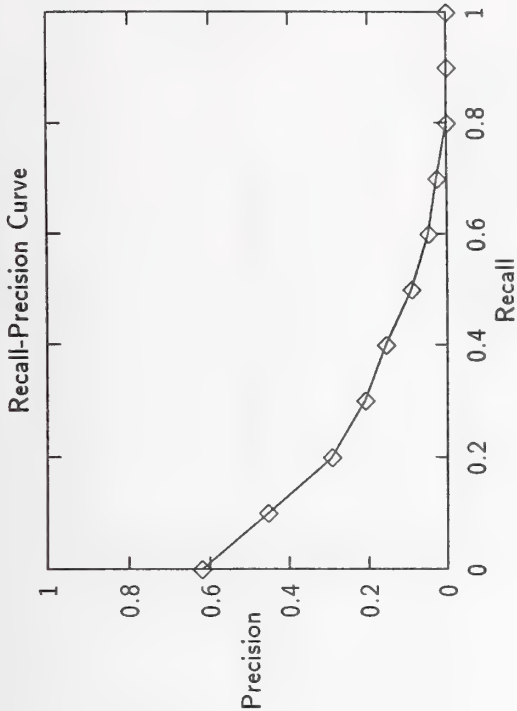
Document Level Averages	
At 5 docs	0.4632
At 10 docs	0.4316
At 15 docs	0.4246
At 20 docs	0.4158
At 30 docs	0.3895
At 100 docs	0.2853
At 200 docs	0.2153
At 500 docs	0.1204
At 1000 docs	0.0656
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3565



Summary Statistics	
Run Number	HIN300-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
RelRet:	542

Recall Level Precision Averages	
Recall	Precision
0.00	0.6195
0.10	0.4539
0.20	0.2944
0.30	0.2094
0.40	0.1570
0.50	0.0929
0.60	0.0475
0.70	0.0250
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.1519

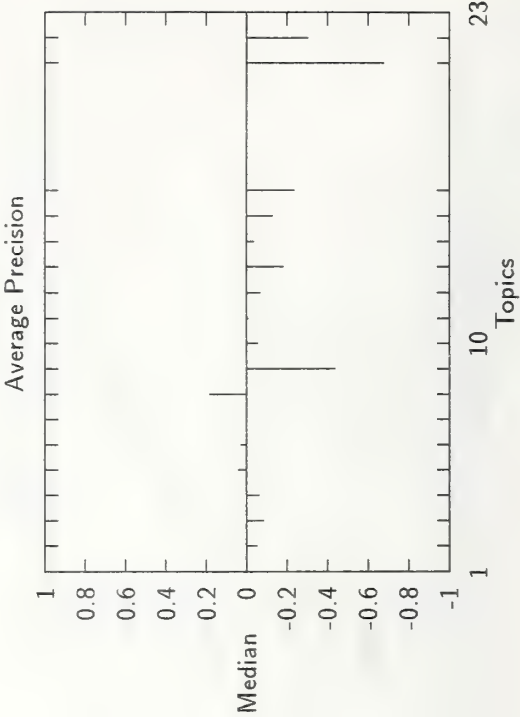
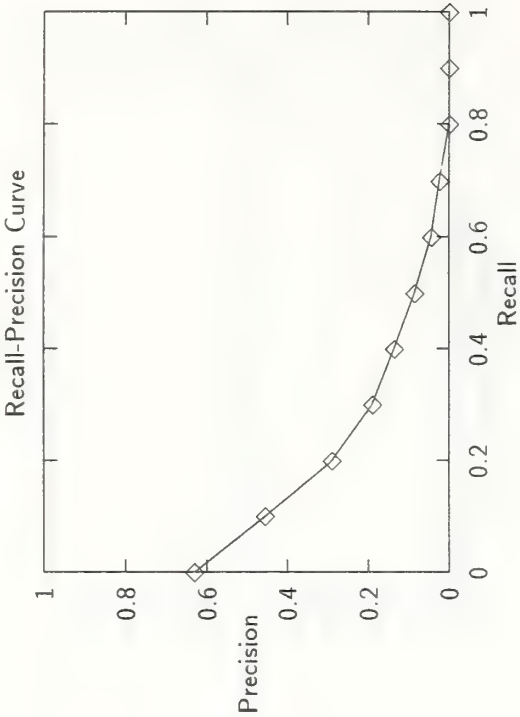
Document Level Averages	
	Precision
At 5 docs	0.4211
At 10 docs	0.3789
At 15 docs	0.3544
At 20 docs	0.3184
At 30 docs	0.2789
At 100 docs	0.1805
At 200 docs	0.1139
At 500 docs	0.0555
At 1000 docs	0.0285
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2333



Summary Statistics	
Run Number	HIN301-automatic
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
Rel_ret:	540

Recall Level Precision Averages	
Recall	Precision
0.00	0.6310
0.10	0.4561
0.20	0.2905
0.30	0.1912
0.40	0.1372
0.50	0.0857
0.60	0.0452
0.70	0.0236
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.1481

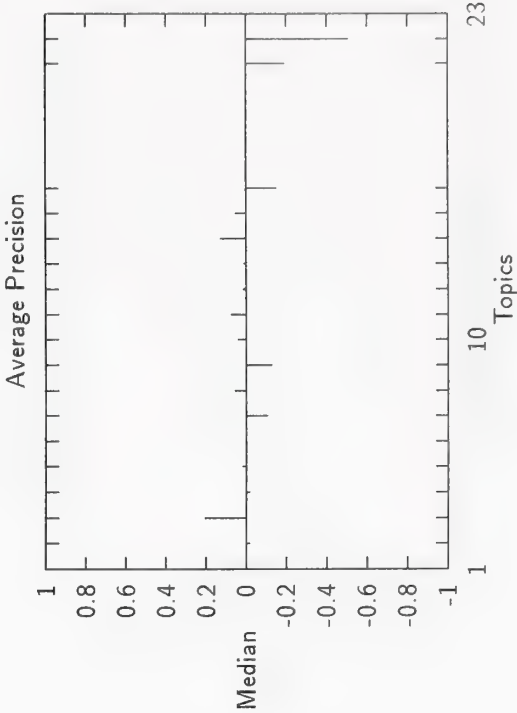
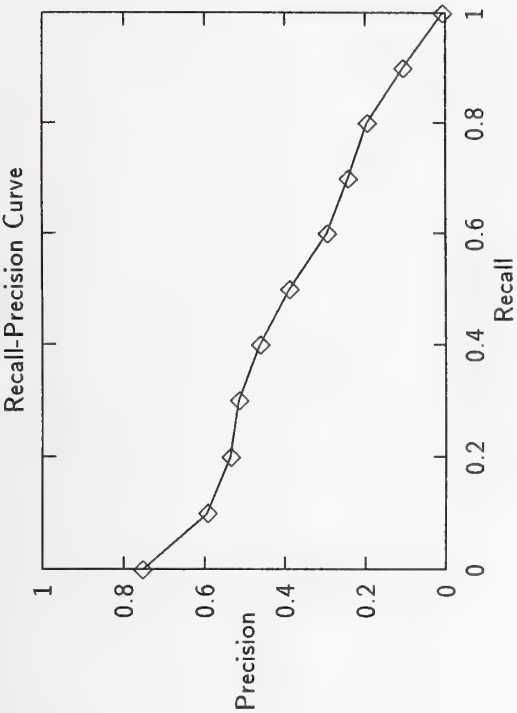
Document Level Averages	
At 5 docs	0.4105
At 10 docs	0.3737
At 15 docs	0.3614
At 20 docs	0.3105
At 30 docs	0.2719
At 100 docs	0.1763
At 200 docs	0.1100
At 500 docs	0.0548
At 1000 docs	0.0284
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2275



Summary Statistics	
Run Number	CLCHNM-manual
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
Rel_ret:	1253

Recall Level Precision Averages	
Recall	Precision
0.00	0.7534
0.10	0.5923
0.20	0.5359
0.30	0.5150
0.40	0.4623
0.50	0.3893
0.60	0.2952
0.70	0.2435
0.80	0.1960
0.90	0.1070
1.00	0.0077
Average precision over all relevant docs	
non-interpolated	0.3583

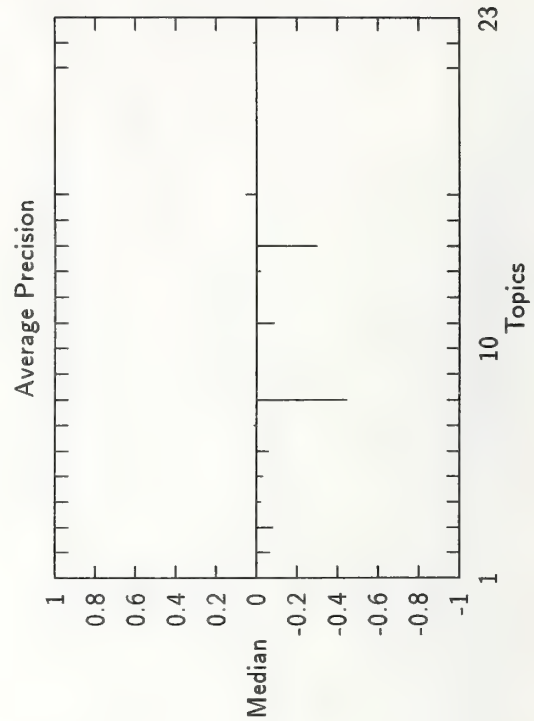
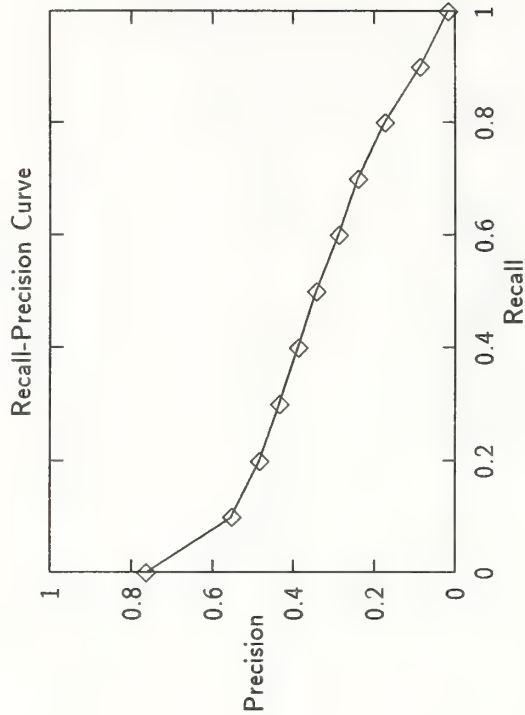
Document Level Averages	
	Precision
At 5 docs	0.5263
At 10 docs	0.5158
At 15 docs	0.5333
At 20 docs	0.4947
At 30 docs	0.4561
At 100 docs	0.3316
At 200 docs	0.2226
At 500 docs	0.1204
At 1000 docs	0.0659
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3872



Summary Statistics	
Run Number	gmu96cm1-manual
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
Rel_ret:	1234

Recall Level Precision Averages	
Recall	Precision
0.00	0.7647
0.10	0.5539
0.20	0.4854
0.30	0.4349
0.40	0.3897
0.50	0.3451
0.60	0.2901
0.70	0.2417
0.80	0.1743
0.90	0.0879
1.00	0.0173
Average precision over all relevant docs	
non-interpolated	0.3279

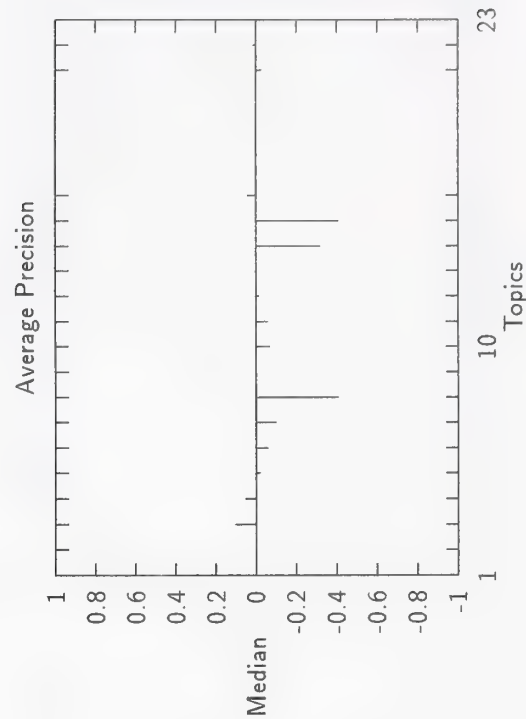
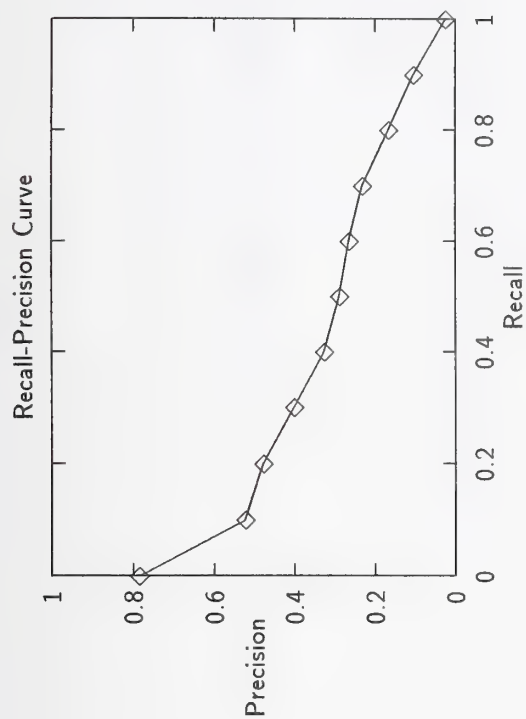
Document Level Averages	
	Precision
At 5 docs	0.5263
At 10 docs	0.4737
At 15 docs	0.4386
At 20 docs	0.4368
At 30 docs	0.4070
At 100 docs	0.2826
At 200 docs	0.2084
At 500 docs	0.1138
At 1000 docs	0.0649
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3594



Summary Statistics	
Run Number	gmu96cm2-manual
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
Rel_ret:	1206

Recall Level Precision Averages	
Recall	Precision
0.00	0.7857
0.10	0.5231
0.20	0.4791
0.30	0.4028
0.40	0.3270
0.50	0.2893
0.60	0.2660
0.70	0.2327
0.80	0.1666
0.90	0.1056
1.00	0.0259
Average precision over all relevant docs	
non-interpolated	0.3065

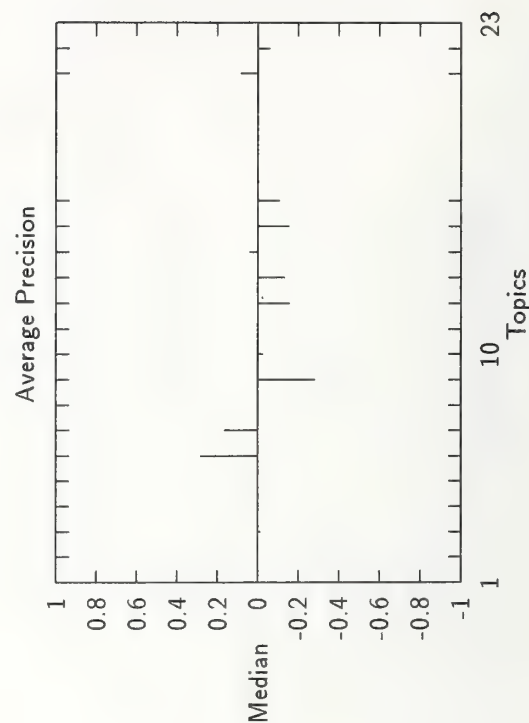
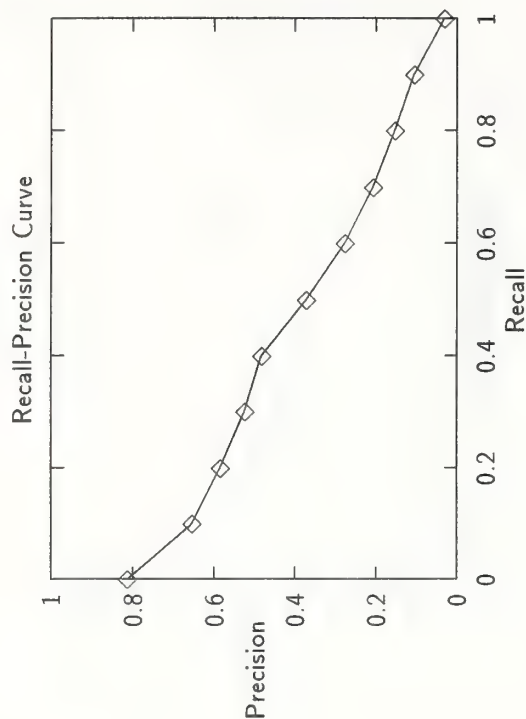
Document Level Averages	
At 5 docs	0.5158
At 10 docs	0.4474
At 15 docs	0.4316
At 20 docs	0.4158
At 30 docs	0.3807
At 100 docs	0.2674
At 200 docs	0.1989
At 500 docs	0.1105
At 1000 docs	0.0635
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3290



Summary Statistics	
Run Number	itcn2-manual
Number of Topics	19
Total number of documents over all topics	
Retrieved:	17501
Relevant:	1399
Rel.ret:	1136

Recall Level Precision Averages	
Recall	Precision
0.00	0.8139
0.10	0.6557
0.20	0.5845
0.30	0.5241
0.40	0.4837
0.50	0.3735
0.60	0.2775
0.70	0.2082
0.80	0.1545
0.90	0.1058
1.00	0.0321
Average precision over all relevant docs	
non-interpolated	0.3607

Document Level Averages	
	Precision
At 5 docs	0.6105
At 10 docs	0.5632
At 15 docs	0.5298
At 20 docs	0.4947
At 30 docs	0.4544
At 100 docs	0.3174
At 200 docs	0.2221
At 500 docs	0.1105
At 1000 docs	0.0598
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3881



Summary Statistics	
Run Number	BrklyCH2-manual
Number of Topics	19
Total number of documents over all topics	
Retrieved:	19000
Relevant:	1399
Rel.ret:	1364

Recall Level Precision Averages	
Recall	Precision
0.00	0.8539
0.10	0.7507
0.20	0.6584
0.30	0.6008
0.40	0.5354
0.50	0.4715
0.60	0.4188
0.70	0.3486
0.80	0.2872
0.90	0.2194
1.00	0.0623
Average precision over all relevant docs	
non-interpolated	0.4610

Document Level Averages	
	Precision
At 5 docs	0.6526
At 10 docs	0.6316
At 15 docs	0.5965
At 20 docs	0.5895
At 30 docs	0.5351
At 100 docs	0.3842
At 200 docs	0.2634
At 500 docs	0.1346
At 1000 docs	0.0718
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4642

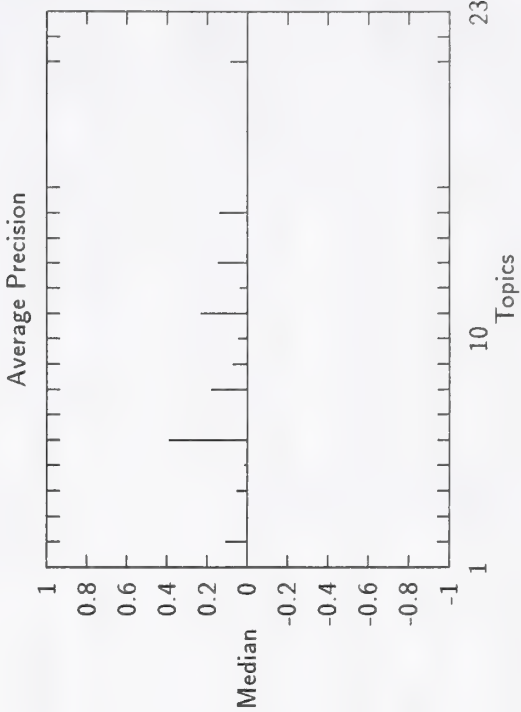
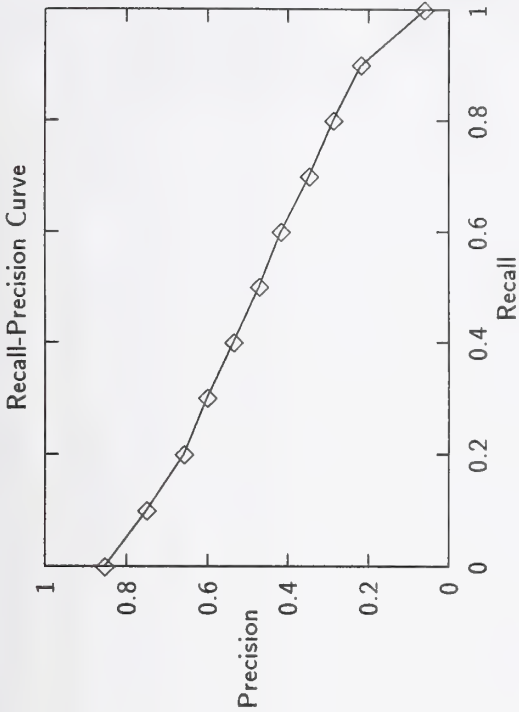


Table 1: Raw Data

topic	num rel	Run 1					Run 2					Run 3				
		set size	pool util.	sample utility			set size	pool util.	sample utility			set size	pool util.	sample utility		
1	30	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
3	101	14	6.0	6.0	+/-	0.0	61	29.0	30.8	+/-	3.7	164	132.0	115.3	+/-	40.9
4	178	9	5.0	5.0	+/-	0.0	10	8.0	8.0	+/-	0.0	10	26.0	26.0	+/-	0.0
5	19	6	-2.0	-2.0	+/-	0.0	14	-2.0	-2.0	+/-	0.0	23	13.0	13.0	+/-	0.0
6	158	2	2.0	2.0	+/-	0.0	7	7.0	7.0	+/-	0.0	52	100.0	100.0	+/-	0.0
11	92	9	-15.0	-15.0	+/-	0.0	23	-7.0	-7.0	+/-	0.0	37	23.0	23.0	+/-	0.0
12	228	7	7.0	7.0	+/-	0.0	34	18.0	18.0	+/-	0.0	54	102.0	102.0	+/-	0.0
23	7	1	1.0	1.0	+/-	0.0	1	1.0	1.0	+/-	0.0	1	3.0	3.0	+/-	0.0
24	38	0	0.0	0.0	+/-	0.0	2	0.0	0.0	+/-	0.0	9	15.0	15.0	+/-	0.0
44	10	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
53	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
54	48	21	-3.0	-3.0	+/-	0.0	50	-2.0	-2.0	+/-	0.0	108	32.0	39.0	+/-	10.0
58	45	1	1.0	1.0	+/-	0.0	8	8.0	8.0	+/-	0.0	8	24.0	24.0	+/-	0.0
68	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0
77	14	3	-1.0	-1.0	+/-	0.0	12	0.0	0.0	+/-	0.0	46	-14.0	-14.0	+/-	0.0
78	37	31	-29.0	-29.0	+/-	0.0	102	-44.0	-47.6	+/-	11.4	133	-1.0	-8.2	+/-	22.7
82	55	33	-15.0	-15.0	+/-	0.0	50	0.0	0.0	+/-	0.0	73	63.0	63.0	+/-	0.0
94	56	2	-6.0	-6.0	+/-	0.0	17	-5.0	-5.0	+/-	0.0	61	-13.0	-13.0	+/-	0.0
95	93	2	2.0	2.0	+/-	0.0	9	5.0	5.0	+/-	0.0	50	38.0	38.0	+/-	0.0
100	157	38	38.0	38.0	+/-	0.0	59	49.0	49.0	+/-	0.0	147	225.0	249.3	+/-	39.8
108	174	121	-131.0	-161.3	+/-	66.3	150	-26.0	-41.2	+/-	33.1	205	111.0	100.7	+/-	69.8
111	887	1000	-728.0	480.0	+/-	251.4	1000	136.0	740.0	+/-	125.7	1000	1272.0	2480.0	+/-	251.4
114	42	0	0.0	0.0	+/-	0.0	4	-2.0	-2.0	+/-	0.0	28	8.0	8.0	+/-	0.0
118	324	11	3.0	3.0	+/-	0.0	37	15.0	15.0	+/-	0.0	125	183.0	180.1	+/-	23.1
119	185	7	-1.0	-1.0	+/-	0.0	7	3.0	3.0	+/-	0.0	240	96.0	110.7	+/-	70.6
121	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0
123	57	3	3.0	3.0	+/-	0.0	4	4.0	4.0	+/-	0.0	22	18.0	18.0	+/-	0.0
125	6	0	0.0	0.0	+/-	0.0	1	1.0	1.0	+/-	0.0	1	3.0	3.0	+/-	0.0
126	18	1	1.0	1.0	+/-	0.0	5	3.0	3.0	+/-	0.0	5	11.0	11.0	+/-	0.0
142	808	65	65.0	65.0	+/-	0.0	150	130.0	139.7	+/-	11.0	443	865.0	1166.3	+/-	126.8
154	22	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	6.0	6.0	+/-	0.0
161	153	9	9.0	9.0	+/-	0.0	26	26.0	26.0	+/-	0.0	49	143.0	143.0	+/-	0.0
173	15	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
185	14	0	0.0	0.0	+/-	0.0	5	-1.0	-1.0	+/-	0.0	12	0.0	0.0	+/-	0.0
187	194	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	4	12.0	12.0	+/-	0.0
189	584	1	1.0	1.0	+/-	0.0	5	5.0	5.0	+/-	0.0	83	173.0	173.0	+/-	0.0
192	10	0	0.0	0.0	+/-	0.0	1	1.0	1.0	+/-	0.0	1	3.0	3.0	+/-	0.0
193	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
194	8	1	-3.0	-3.0	+/-	0.0	2	0.0	0.0	+/-	0.0	3	1.0	1.0	+/-	0.0
195	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
202	583	187	-65.0	33.0	+/-	93.3	491	43.0	248.2	+/-	100.1	835	653.0	1393.9	+/-	301.7
207	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
211	2	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
221	193	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	6.0	6.0	+/-	0.0
222	2	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0
224	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0
228	68	0	0.0	0.0	+/-	0.0	5	1.0	1.0	+/-	0.0	5	7.0	7.0	+/-	0.0
240	88	2	-6.0	-6.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0	66	-10.0	-10.0	+/-	0.0
243	2	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0

Summary Statistics

Run Number city96f-category A, automatic
Number of Topics 49

Total number of documents over all topics

	Run 1	Run 2	Run 3
Retrieved:	1587	2355	4118
Relevant:	5808	5808	5808
Rel_ret:	975	1378	2109

Average Effectiveness
(based on pool of submitted docs,
and averaged over 45 topics)

	Run 1	Run 2	Run 3
Mean Precision:	0.4338	0.4870	0.4794
Mean Recall:	0.0809	0.1535	0.2435

Table 1: Raw Data

topic	num rel	Run 1					Run 2					Run 3				
		set size	pool util.	sample utility			set size	pool util.	sample utility			set size	pool util.	sample utility		
1	30	0	0.0	0.0	+/	0.0	1	-1.0	-1.0	+/	0.0	6	6.0	6.0	+/	0.0
3	101	6	-2.0	-2.0	+/	0.0	11	3.0	3.0	+/	0.0	76	124.0	124.0	+/	0.0
4	178	14	14.0	14.0	+/	0.0	14	14.0	14.0	+/	0.0	46	130.0	130.0	+/	0.0
5	19	5	-7.0	-7.0	+/	0.0	6	-2.0	-2.0	+/	0.0	24	12.0	12.0	+/	0.0
6	158	0	0.0	0.0	+/	0.0	10	10.0	10.0	+/	0.0	27	77.0	77.0	+/	0.0
11	92	16	-20.0	-20.0	+/	0.0	20	-2.0	-2.0	+/	0.0	39	29.0	29.0	+/	0.0
12	228	23	-9.0	-9.0	+/	0.0	52	8.0	8.0	+/	0.0	90	142.0	142.0	+/	0.0
23	7	1	1.0	1.0	+/	0.0	1	1.0	1.0	+/	0.0	1	3.0	3.0	+/	0.0
24	38	2	2.0	2.0	+/	0.0	3	1.0	1.0	+/	0.0	4	8.0	8.0	+/	0.0
44	10	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0
53	1	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0
54	48	6	2.0	2.0	+/	0.0	27	15.0	15.0	+/	0.0	102	42.0	43.6	+/	4.6
58	45	0	0.0	0.0	+/	0.0	1	1.0	1.0	+/	0.0	7	21.0	21.0	+/	0.0
68	0	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0
77	14	1	-3.0	-3.0	+/	0.0	30	-18.0	-18.0	+/	0.0	66	-34.0	-34.0	+/	0.0
78	37	11	7.0	7.0	+/	0.0	55	-11.0	-11.0	+/	0.0	93	15.0	15.0	+/	0.0
82	55	18	6.0	6.0	+/	0.0	38	-2.0	-2.0	+/	0.0	56	56.0	56.0	+/	0.0
94	56	0	0.0	0.0	+/	0.0	1	-1.0	-1.0	+/	0.0	9	3.0	3.0	+/	0.0
95	93	0	0.0	0.0	+/	0.0	3	-1.0	-1.0	+/	0.0	51	-7.0	-7.0	+/	0.0
100	157	18	18.0	18.0	+/	0.0	77	61.0	61.1	+/	3.8	107	225.0	225.2	+/	7.6
108	174	12	0.0	0.0	+/	0.0	135	-13.0	11.6	+/	29.4	327	85.0	175.6	+/	107.3
111	887	766	-466.0	675.9	+/	172.7	1443	-105.0	1192.8	+/	185.4	1547	1285.0	4077.5	+/	373.2
114	42	0	0.0	0.0	+/	0.0	2	0.0	0.0	+/	0.0	45	-1.0	-1.0	+/	0.0
118	324	10	6.0	6.0	+/	0.0	37	25.0	25.0	+/	0.0	112	208.0	202.5	+/	14.4
119	185	0	0.0	0.0	+/	0.0	39	3.0	3.0	+/	0.0	327	9.0	2.5	+/	106.0
121	0	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0
123	57	0	0.0	0.0	+/	0.0	5	1.0	1.0	+/	0.0	10	14.0	14.0	+/	0.0
125	6	1	1.0	1.0	+/	0.0	1	1.0	1.0	+/	0.0	3	5.0	5.0	+/	0.0
126	18	6	2.0	2.0	+/	0.0	7	3.0	3.0	+/	0.0	10	22.0	22.0	+/	0.0
142	808	44	40.0	38.8	+/	4.8	137	129.0	134.4	+/	2.4	340	792.0	990.2	+/	44.4
154	22	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0	5	11.0	11.0	+/	0.0
161	153	8	8.0	8.0	+/	0.0	28	28.0	28.0	+/	0.0	49	147.0	147.0	+/	0.0
173	15	1	-3.0	-3.0	+/	0.0	1	-1.0	-1.0	+/	0.0	2	2.0	2.0	+/	0.0
185	14	1	1.0	1.0	+/	0.0	3	1.0	1.0	+/	0.0	7	9.0	9.0	+/	0.0
187	194	2	2.0	2.0	+/	0.0	3	1.0	1.0	+/	0.0	16	36.0	36.0	+/	0.0
189	584	0	0.0	0.0	+/	0.0	11	-1.0	-1.0	+/	0.0	53	75.0	75.0	+/	0.0
192	10	1	1.0	1.0	+/	0.0	1	1.0	1.0	+/	0.0	1	3.0	3.0	+/	0.0
193	0	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0
194	8	1	-3.0	-3.0	+/	0.0	2	-2.0	-2.0	+/	0.0	2	-2.0	-2.0	+/	0.0
195	0	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0
202	583	294	-194.0	-51.9	+/	171.9	625	-49.0	171.2	+/	137.7	900	644.0	1192.4	+/	327.9
207	1	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0
211	2	0	0.0	0.0	+/	0.0	1	-1.0	-1.0	+/	0.0	1	-1.0	-1.0	+/	0.0
221	193	2	2.0	2.0	+/	0.0	2	2.0	2.0	+/	0.0	9	15.0	15.0	+/	0.0
222	2	0	0.0	0.0	+/	0.0	3	-3.0	-3.0	+/	0.0	3	-3.0	-3.0	+/	0.0
224	1	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0	1	-1.0	-1.0	+/	0.0
228	68	6	-6.0	-6.0	+/	0.0	6	0.0	0.0	+/	0.0	16	12.0	12.0	+/	0.0
240	88	0	0.0	0.0	+/	0.0	2	-2.0	-2.0	+/	0.0	33	-5.0	-5.0	+/	0.0
243	2	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0	0	0.0	0.0	+/	0.0

Summary Statistics

Run Number	INR3—category A, automatic		
Number of Topics	49		
Total number of documents over all topics			
	Run 1	Run 2	Run 3
Retrieved:	1276	2844	4623
Relevant:	5808	5808	5808
Rel_ret:	807	1469	2209

Average Effectiveness
(based on pool of submitted docs,
and averaged over 45 topics)

	Run 1	Run 2	Run 3
Mean Precision:	0.4417	0.4793	0.4843
Mean Recall:	0.0667	0.1332	0.2466

Table 1: Raw Data

topic	num rel	Run 1					Run 2					Run 3				
		set size	pool util.	sample utility			set size	pool util.	sample utility			set size	pool util.	sample utility		
1	30	4	-12.0	-12.0	+/-	0.0	27	-25.0	-25.0	+/-	0.0	63	-59.0	-59.0	+/-	0.0
3	101	7	3.0	3.0	+/-	0.0	63	-25.0	-20.0	+/-	5.8	495	-315.0	-333.9	+/-	98.0
4	178	2	2.0	2.0	+/-	0.0	16	-4.0	-4.0	+/-	0.0	176	48.0	23.2	+/-	42.3
5	19	11	-25.0	-25.0	+/-	0.0	49	-35.0	-35.0	+/-	0.0	88	-44.0	-44.0	+/-	0.0
6	158	8	-12.0	-12.0	+/-	0.0	46	-38.0	-38.0	+/-	0.0	120	-100.0	-98.5	+/-	5.6
11	92	3	3.0	3.0	+/-	0.0	37	-3.0	-3.0	+/-	0.0	106	14.0	5.8	+/-	7.4
12	228	2	-2.0	-2.0	+/-	0.0	57	-15.0	-16.8	+/-	4.9	409	-17.0	131.1	+/-	173.6
23	7	5	-11.0	-11.0	+/-	0.0	58	-56.0	-56.0	+/-	0.0	58	-54.0	-54.0	+/-	0.0
24	38	8	-20.0	-20.0	+/-	0.0	59	-47.0	-47.0	+/-	0.0	95	-55.0	-55.0	+/-	0.0
44	10	2	-6.0	-6.0	+/-	0.0	46	-46.0	-46.0	+/-	0.0	173	-173.0	-173.0	+/-	0.0
53	1	15	-45.0	-45.0	+/-	0.0	78	-78.0	-78.0	+/-	0.0	138	-134.0	-132.3	+/-	6.1
54	48	11	-9.0	-9.0	+/-	0.0	60	-32.0	-32.8	+/-	3.0	202	-66.0	-44.2	+/-	54.7
58	45	17	-23.0	-23.0	+/-	0.0	117	-79.0	-79.2	+/-	15.1	237	-113.0	-91.1	+/-	52.3
68	0	2	-6.0	-6.0	+/-	0.0	6	-6.0	-6.0	+/-	0.0	23	-23.0	-23.0	+/-	0.0
77	14	2	-6.0	-6.0	+/-	0.0	11	-9.0	-9.0	+/-	0.0	40	-28.0	-28.0	+/-	0.0
78	37	4	4.0	4.0	+/-	0.0	16	2.0	2.0	+/-	0.0	47	25.0	25.0	+/-	0.0
82	55	3	-5.0	-5.0	+/-	0.0	9	-7.0	-7.0	+/-	0.0	23	-3.0	-3.0	+/-	0.0
94	56	1	-3.0	-3.0	+/-	0.0	9	-3.0	-3.0	+/-	0.0	35	1.0	1.0	+/-	0.0
95	93	4	0.0	0.0	+/-	0.0	10	2.0	2.0	+/-	0.0	22	14.0	14.0	+/-	0.0
100	157	4	0.0	0.0	+/-	0.0	67	-27.0	-32.1	+/-	7.4	315	-71.0	-121.3	+/-	85.5
108	174	6	-14.0	-14.0	+/-	0.0	107	-53.0	-44.8	+/-	19.5	404	-136.0	-52.2	+/-	129.9
111	887	94	-62.0	38.7	+/-	36.3	656	-92.0	424.0	+/-	146.9	1594	674.0	3067.3	+/-	669.8
114	42	11	-29.0	-29.0	+/-	0.0	45	-37.0	-37.0	+/-	0.0	62	-42.0	-42.0	+/-	0.0
118	324	18	-30.0	-30.0	+/-	0.0	189	-109.0	-85.2	+/-	40.9	565	-205.0	46.0	+/-	211.4
119	185	4	-4.0	-4.0	+/-	0.0	86	-48.0	-51.2	+/-	11.8	655	-331.0	78.3	+/-	284.0
121	0	9	-27.0	-27.0	+/-	0.0	49	-49.0	-49.0	+/-	0.0	49	-49.0	-49.0	+/-	0.0
123	57	5	-11.0	-11.0	+/-	0.0	35	-7.0	-7.0	+/-	0.0	67	17.0	17.0	+/-	0.0
125	6	4	-12.0	-12.0	+/-	0.0	38	-36.0	-36.0	+/-	0.0	59	-55.0	-55.0	+/-	0.0
126	18	2	-6.0	-6.0	+/-	0.0	8	2.0	2.0	+/-	0.0	9	11.0	11.0	+/-	0.0
142	808	15	15.0	15.0	+/-	0.0	76	62.0	67.5	+/-	5.1	164	380.0	424.7	+/-	29.1
154	22	6	-6.0	-6.0	+/-	0.0	17	-7.0	-7.0	+/-	0.0	17	3.0	3.0	+/-	0.0
161	153	10	10.0	10.0	+/-	0.0	41	33.0	33.0	+/-	0.0	48	128.0	128.0	+/-	0.0
173	15	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
185	14	5	-15.0	-15.0	+/-	0.0	64	-56.0	-54.2	+/-	4.0	262	-226.0	-242.3	+/-	8.1
187	194	4	-12.0	-12.0	+/-	0.0	123	-95.0	-98.2	+/-	16.1	388	-256.0	-250.1	+/-	82.3
189	584	1	1.0	1.0	+/-	0.0	29	5.0	5.0	+/-	0.0	143	185.0	220.4	+/-	31.3
192	10	8	-12.0	-12.0	+/-	0.0	18	-6.0	-6.0	+/-	0.0	18	6.0	6.0	+/-	0.0
193	0	3	-9.0	-9.0	+/-	0.0	19	-19.0	-19.0	+/-	0.0	19	-19.0	-19.0	+/-	0.0
194	8	7	-21.0	-21.0	+/-	0.0	95	-93.0	-92.1	+/-	3.1	127	-123.0	-121.2	+/-	6.3
195	0	4	-12.0	-12.0	+/-	0.0	19	-19.0	-19.0	+/-	0.0	19	-19.0	-19.0	+/-	0.0
202	583	3	-5.0	-5.0	+/-	0.0	83	-31.0	-22.2	+/-	13.6	445	123.0	98.9	+/-	177.3
207	1	3	-5.0	-5.0	+/-	0.0	20	-18.0	-18.0	+/-	0.0	20	-16.0	-16.0	+/-	0.0
211	2	5	-11.0	-11.0	+/-	0.0	7	-5.0	-5.0	+/-	0.0	7	-3.0	-3.0	+/-	0.0
221	193	8	-24.0	-24.0	+/-	0.0	65	-63.0	-65.0	+/-	0.0	243	-207.0	-212.0	+/-	36.5
222	2	3	-9.0	-9.0	+/-	0.0	37	-37.0	-37.0	+/-	0.0	49	-49.0	-49.0	+/-	0.0
224	1	5	-15.0	-15.0	+/-	0.0	13	-13.0	-13.0	+/-	0.0	13	-13.0	-13.0	+/-	0.0
228	68	8	-24.0	-24.0	+/-	0.0	42	-36.0	-36.0	+/-	0.0	86	-58.0	-58.0	+/-	0.0
240	88	20	-44.0	-44.0	+/-	0.0	232	-186.0	-192.2	+/-	31.6	954	-778.0	-874.4	+/-	63.1
243	2	7	-17.0	-17.0	+/-	0.0	72	-68.0	-70.0	+/-	0.0	216	-208.0	-212.0	+/-	0.0

Summary Statistics

Run Number	INTXA-category A, automatic		
Number of Topics	49		
Total number of documents over all topics			
	Run 1	Run 2	Run 3
Retrieved:	393	3126	9567
Relevant:	5808	5808	5808
Rel_ret:	149	757	1787

Average Effectiveness
(based on pool of submitted docs,
and averaged over 45 topics)

	Run 1	Run 2	Run 3
Mean Precision:	0.3331	0.2326	0.1913
Mean Recall:	0.0793	0.1894	0.3295

Table 1: Raw Data

topic	num rel	Run 1						Run 2						Run 3					
		set size	pool util.	sample utility				set size	pool util.	sample utility				set size	pool util.	sample utility			
1	30	3	-9.0	-9.0	+/	0.0		79	-77.0	-77.0	+/	0.0		79	-75.0	-75.0	+/	0.0	
3	101	21	-11.0	-11.0	+/	0.0		529	-427.0	-452.2	+/	66.7		598	-386.0	-430.2	+/	134.0	
4	178	3	3.0	3.0	+/	0.0		328	-250.0	-248.3	+/	34.5		328	-172.0	-168.6	+/	69.1	
5	19	6	-18.0	-18.0	+/	0.0		80	-64.0	-64.0	+/	0.0		80	-48.0	-48.0	+/	0.0	
6	158	4	0.0	0.0	+/	0.0		48	-36.0	-36.0	+/	0.0		48	-24.0	-24.0	+/	0.0	
11	92	3	-1.0	-1.0	+/	0.0		88	-20.0	-20.0	+/	0.0		88	48.0	48.0	+/	0.0	
12	228	11	-5.0	-5.0	+/	0.0		412	-176.0	-98.9	+/	92.2		442	58.0	212.1	+/	184.4	
23	7	1	1.0	1.0	+/	0.0		17	-15.0	-15.0	+/	0.0		18	-14.0	-14.0	+/	0.0	
24	38	43	-97.0	-97.0	+/	0.0		454	-404.0	-380.3	+/	51.0		454	-354.0	-306.6	+/	102.1	
44	10	54	-158.0	-162.0	+/	0.0		676	-666.0	-638.3	+/	71.9		769	-749.0	-693.6	+/	143.8	
53	1	6	-18.0	-18.0	+/	0.0		22	-22.0	-22.0	+/	0.0		22	-22.0	-22.0	+/	0.0	
54	48	12	-20.0	-20.0	+/	0.0		283	-209.0	-262.7	+/	22.1		331	-183.0	-290.4	+/	44.2	
58	45	15	-9.0	-9.0	+/	0.0		375	-297.0	-341.0	+/	29.3		415	-259.0	-347.0	+/	58.7	
68	0	3	-9.0	-9.0	+/	0.0		64	-64.0	-64.0	+/	0.0		77	-77.0	-77.0	+/	0.0	
77	14	7	-13.0	-13.0	+/	0.0		95	-79.0	-79.0	+/	0.0		97	-65.0	-65.0	+/	0.0	
78	37	19	-17.0	-17.0	+/	0.0		57	-13.0	-13.0	+/	0.0		58	30.0	30.0	+/	0.0	
82	55	30	-50.0	-50.0	+/	0.0		220	-146.0	-165.5	+/	21.5		224	-76.0	-114.9	+/	42.9	
94	56	10	-18.0	-18.0	+/	0.0		49	-27.0	-27.0	+/	0.0		57	3.0	3.0	+/	0.0	
95	93	13	-19.0	-19.0	+/	0.0		50	-2.0	-2.0	+/	0.0		50	46.0	46.0	+/	0.0	
100	157	8	-12.0	-12.0	+/	0.0		215	-125.0	-141.0	+/	30.4		240	-60.0	-92.1	+/	60.9	
108	174	4	-4.0	-4.0	+/	0.0		456	-348.0	-318.8	+/	57.6		457	-241.0	-182.6	+/	115.1	
111	887	64	0.0	26.4	+/	21.2		1591	-501.0	831.8	+/	448.7		1698	658.0	3497.8	+/	898.6	
114	42	74	-186.0	-191.8	+/	14.7		214	-180.0	-187.2	+/	14.9		217	-149.0	-163.5	+/	29.8	
118	324	1772	-4728.0	-4065.2	+/	913.0		4915	-4511.0	-4099.1	+/	588.5		5208	-4388.0	-3576.2	+/	1177.0	
119	185	1772	-4816.0	-4065.2	+/	913.0		4915	-4587.0	-4099.1	+/	588.5		5208	-4536.0	-3505.2	+/	1130.5	
121	0	37	-111.0	-111.0	+/	0.0		110	-110.0	-110.0	+/	0.0		113	-113.0	-113.0	+/	0.0	
123	57	55	-141.0	-146.3	+/	6.8		247	-177.0	-187.6	+/	33.1		254	-110.0	-131.1	+/	66.2	
125	6	1	-3.0	-3.0	+/	0.0		28	-28.0	-28.0	+/	0.0		31	-31.0	-31.0	+/	0.0	
126	18	1	-3.0	-3.0	+/	0.0		10	2.0	2.0	+/	0.0		14	14.0	14.0	+/	0.0	
142	808	5	5.0	5.0	+/	0.0		145	63.0	86.1	+/	13.1		140	256.0	302.1	+/	26.2	
154	22	4	0.0	0.0	+/	0.0		32	-16.0	-16.0	+/	0.0		34	2.0	2.0	+/	0.0	
161	153	7	3.0	3.0	+/	0.0		40	30.0	30.0	+/	0.0		55	145.0	145.0	+/	0.0	
173	15	1	-3.0	-3.0	+/	0.0		1	-1.0	-1.0	+/	0.0		1	-1.0	-1.0	+/	0.0	
185	14	88	-252.0	-253.6	+/	15.9		747	-723.0	-741.8	+/	7.9		780	-732.0	-769.6	+/	15.9	
187	194	81	-203.0	-197.6	+/	19.5		1399	-1197.0	-1060.0	+/	235.5		1399	-995.0	-721.0	+/	470.9	
189	584	400	-696.0	-147.4	+/	232.7		1350	-694.0	357.4	+/	317.2		1375	5.0	2107.8	+/	634.4	
192	10	2	2.0	2.0	+/	0.0		25	-15.0	-15.0	+/	0.0		25	-5.0	-5.0	+/	0.0	
193	0	3	-9.0	-9.0	+/	0.0		45	-45.0	-45.0	+/	0.0		45	-45.0	-45.0	+/	0.0	
194	8	617	-1835.0	-1851.0	+/	0.0		4307	-4297.0	-4307.0	+/	0.0		4755	-4735.0	-4755.0	+/	0.0	
195	0	6	-18.0	-18.0	+/	0.0		24	-24.0	-24.0	+/	0.0		24	-24.0	-24.0	+/	0.0	
202	583	11	3.0	3.0	+/	0.0		162	6.0	47.5	+/	21.4		169	195.0	278.0	+/	42.8	
207	1	6	-18.0	-18.0	+/	0.0		42	-40.0	-40.0	+/	0.0		64	-60.0	-60.0	+/	0.0	
211	2	7	-21.0	-21.0	+/	0.0		29	-25.0	-25.0	+/	0.0		29	-21.0	-21.0	+/	0.0	
221	193	23	-57.0	-57.0	+/	0.0		332	-296.0	-262.6	+/	55.7		1041	-929.0	-753.0	+/	227.7	
222	2	13	-39.0	-39.0	+/	0.0		98	-98.0	-98.0	+/	0.0		101	-101.0	-101.0	+/	0.0	
224	1	4	-12.0	-12.0	+/	0.0		9	-9.0	-9.0	+/	0.0		9	-9.0	-9.0	+/	0.0	
228	68	21	-55.0	-55.0	+/	0.0		260	-218.0	-180.5	+/	39.8		282	-194.0	-119.1	+/	79.7	
240	88	158	-406.0	-343.9	+/	77.2		1669	-1543.0	-1329.2	+/	300.2		1778	-1510.0	-1072.0	+/	601.2	
243	2	3	-9.0	-9.0	+/	0.0		177	-175.0	-170.4	+/	10.7		221	-217.0	-207.9	+/	21.5	

Summary Statistics

Run Number INTXM—category A, manual
Number of Topics 49

Total number of documents over all topics

	Run 1	Run 2	Run 3
Retrieved:	5512	27520	30002
Relevant:	5808	5808	5808
Rel_ret:	611	2322	2438

Average Effectiveness
(based on pool of submitted docs,
and averaged over 45 topics)

	Run 1	Run 2	Run 3
Mean Precision:	0.3308	0.1679	0.1654
Mean Recall:	0.1068	0.4212	0.4338

topic	num rel	Run 1					Run 2					Run 3				
		set size	pool util.	sample utility			set size	pool util.	sample utility			set size	pool util.	sample utility		
1	30	25	-67.0	-67.0	+/-	0.0	250	-232.0	-246.0	+/-	0.0	500	-448.0	-437.9	+/-	68.2
3	101	25	-59.0	-59.0	+/-	0.0	250	-222.0	-242.0	+/-	0.0	500	-424.0	-484.0	+/-	0.0
4	178	25	-3.0	-3.0	+/-	0.0	250	-98.0	-107.4	+/-	56.2	500	-140.0	-160.8	+/-	131.5
5	19	25	-67.0	-67.0	+/-	0.0	250	-228.0	-234.2	+/-	21.2	500	-448.0	-468.3	+/-	42.3
6	158	25	-31.0	-31.0	+/-	0.0	250	-198.0	-204.3	+/-	29.5	500	-344.0	-354.6	+/-	90.2
11	92	25	-27.0	-27.0	+/-	0.0	250	-134.0	-178.6	+/-	40.6	500	-200.0	-303.2	+/-	106.0
12	228	25	-31.0	-31.0	+/-	0.0	250	-46.0	-26.7	+/-	65.7	500	48.0	135.8	+/-	176.7
23	7	25	-63.0	-63.0	+/-	0.0	250	-244.0	-244.0	+/-	0.0	500	-484.0	-488.0	+/-	0.0
24	38	25	-51.0	-51.0	+/-	0.0	250	-228.0	-226.2	+/-	21.2	500	-456.0	-452.3	+/-	42.3
44	10	25	-71.0	-71.0	+/-	0.0	250	-246.0	-236.2	+/-	21.2	500	-480.0	-472.3	+/-	42.3
53	1	25	-75.0	-75.0	+/-	0.0	250	-248.0	-250.0	+/-	0.0	500	-496.0	-500.0	+/-	0.0
54	48	25	-43.0	-43.0	+/-	0.0	250	-190.0	-174.8	+/-	44.7	500	-356.0	-322.6	+/-	101.9
58	45	25	-63.0	-63.0	+/-	0.0	250	-234.0	-220.3	+/-	29.5	500	-460.0	-440.6	+/-	59.0
68	0	25	-75.0	-75.0	+/-	0.0	250	-250.0	-250.0	+/-	0.0	500	-500.0	-500.0	+/-	0.0
77	14	25	-55.0	-55.0	+/-	0.0	250	-238.0	-240.0	+/-	0.0	500	-472.0	-480.0	+/-	0.0
78	37	25	-63.0	-63.0	+/-	0.0	250	-224.0	-196.6	+/-	40.6	500	-432.0	-339.2	+/-	106.0
82	55	25	-23.0	-23.0	+/-	0.0	250	-162.0	-117.4	+/-	56.2	500	-308.0	-234.8	+/-	112.4
94	56	25	-75.0	-75.0	+/-	0.0	250	-242.0	-250.0	+/-	0.0	500	-480.0	-500.0	+/-	0.0
95	93	25	-43.0	-43.0	+/-	0.0	250	-222.0	-210.3	+/-	29.5	500	-424.0	-393.6	+/-	76.7
100	157	25	1.0	1.0	+/-	0.0	250	-140.0	-129.1	+/-	51.2	500	-228.0	-150.1	+/-	138.8
108	174	25	-39.0	-39.0	+/-	0.0	250	-166.0	-172.8	+/-	44.7	500	-232.0	-156.4	+/-	148.1
111	887	25	17.0	17.0	+/-	0.0	250	44.0	32.8	+/-	66.0	500	576.0	498.1	+/-	199.3
114	42	25	-59.0	-59.0	+/-	0.0	250	-226.0	-230.2	+/-	21.2	500	-448.0	-460.3	+/-	42.3
118	324	25	-75.0	-75.0	+/-	0.0	250	-242.0	-250.0	+/-	0.0	500	-448.0	-418.9	+/-	82.3
119	185	25	-71.0	-71.0	+/-	0.0	250	-242.0	-236.2	+/-	21.2	500	-468.0	-472.3	+/-	42.3
121	0	25	-75.0	-75.0	+/-	0.0	250	-250.0	-250.0	+/-	0.0	500	-500.0	-500.0	+/-	0.0
123	57	25	-55.0	-55.0	+/-	0.0	250	-222.0	-228.2	+/-	21.2	500	-436.0	-456.3	+/-	42.3
125	6	25	-75.0	-75.0	+/-	0.0	250	-248.0								

Summary Statistics

Run Number	ispF-category A, automatic
Number of Topics	49

Total number of documents over all topics

	Run 1	Run 2	Run 3
Retrieved:	1224	12249	24499
Relevant:	5808	5808	5808
Rel_ret:	264	1198	1774

Average Effectiveness
(based on pool of submitted docs,
and averaged over 45 topics)

	Run 1	Run 2	Run 3
Mean Precision:	0.2347	0.1065	0.0788
Mean Recall:	0.0958	0.3008	0.3716

Table 1: Raw Data

topic	num rel	Run 1			Run 2			Run 3		
		set size	pool util.	sample utility	set size	pool util.	sample utility	set size	pool util.	sample utility
1	30	0	0.0	0.0 +/- 0.0	3	-3.0	-3.0 +/- 0.0	54	-38.0	-38.0 +/- 0.0
3	101	26	-42.0	-42.0 +/- 0.0	166	-86.0	-72.3 +/- 34.8	662	-398.0	-421.0 +/- 122.8
4	178	33	29.0	29.0 +/- 0.0	89	51.0	44.2 +/- 11.6	143	189.0	162.6 +/- 29.4
5	19	19	-53.0	-53.0 +/- 0.0	27	-25.0	-25.0 +/- 0.0	29	-25.0	-25.0 +/- 0.0
6	158	98	-218.0	-236.4 +/- 38.3	186	-122.0	-151.8 +/- 20.8	679	-407.0	-491.2 +/- 162.9
11	92	14	-6.0	-6.0 +/- 0.0	80	-20.0	-25.2 +/- 10.6	487	-171.0	-36.6 +/- 190.6
12	228	43	-21.0	-12.6 +/- 12.6	265	-75.0	-45.4 +/- 68.6	559	-11.0	379.2 +/- 234.1
23	7	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0	25	-21.0	-21.0 +/- 0.0
24	38	84	-244.0	-242.1 +/- 14.9	134	-128.0	-126.0 +/- 8.2	977	-917.0	-858.9 +/- 197.0
44	10	10	-26.0	-26.0 +/- 0.0	28	-26.0	-26.0 +/- 0.0	140	-136.0	-136.0 +/- 0.0
53	1	1	-3.0	-3.0 +/- 0.0	1	-1.0	-1.0 +/- 0.0	1	-1.0	-1.0 +/- 0.0
54	48	64	-84.0	-84.0 +/- 0.0	96	-36.0	-36.0 +/- 0.0	96	24.0	24.0 +/- 0.0
58	45	197	-463.0	-405.6 +/- 103.7	236	-168.0	-138.6 +/- 51.9	384	-224.0	-171.2 +/- 108.4
68	0	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0
77	14	57	-155.0	-150.9 +/- 14.0	136	-124.0	-121.2 +/- 10.0	229	-193.0	-188.0 +/- 26.8
78	37	81	-167.0	-138.2 +/- 39.4	137	-95.0	-84.6 +/- 19.7	214	-126.0	-109.2 +/- 39.4
82	55	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0
94	56	5	-11.0	-11.0 +/- 0.0	62	-54.0	-54.0 +/- 0.0	100	-80.0	-80.0 +/- 0.0
95	93	1	-3.0	-3.0 +/- 0.0	24	-18.0	-18.0 +/- 0.0	437	-361.0	-359.8 +/- 65.8
100	157	98	-110.0	-74.9 +/- 53.7	274	-130.0	-100.5 +/- 50.2	419	-91.0	-36.8 +/- 108.8
108	174	57	-95.0	-103.9 +/- 22.5	336	-216.0	-217.9 +/- 66.1	598	-278.0	-171.3 +/- 186.0
111	887	1627	-2865.0	-1818.4 +/- 1096.7	2043	-881.0	17.7 +/- 564.3	4062	-802.0	4954.0 +/- 1764.2
114	42	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0
118	324	15	-9.0	-9.0 +/- 0.0	447	-295.0	-348.6 +/- 72.0	1871	-1167.0	-1267.4 +/- 465.2
119	185	0	0.0	0.0 +/- 0.0	276	-220.0	-220.8 +/- 42.0	1738	-1398.0	-1393.7 +/- 326.3
121	0	8	-24.0	-24.0 +/- 0.0	10	-10.0	-10.0 +/- 0.0	53	-53.0	-53.0 +/- 0.0
123	57	9	-7.0	-7.0 +/- 0.0	33	-21.0	-21.0 +/- 0.0	96	-36.0	-36.0 +/- 0.0
125	6	0	0.0	0.0 +/- 0.0	11	-7.0	-7.0 +/- 0.0	22	-14.0	-14.0 +/- 0.0
126	18	9	1.0	1.0 +/- 0.0	21	-3.0	-3.0 +/- 0.0	62	-22.0	-22.0 +/- 0.0
142	808	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0	937	755.0	1649.1 +/- 322.7
154	22	26	-54.0	-54.0 +/- 0.0	84	-70.0	-68.9 +/- 3.7	582	-514.0	-551.7 +/- 7.4
161	153	19	15.0	15.0 +/- 0.0	40	20.0	20.0 +/- 0.0	49	91.0	91.0 +/- 0.0
173	15	1	-3.0	-3.0 +/- 0.0	37	-37.0	-37.0 +/- 0.0	86	-86.0	-86.0 +/- 0.0
185	14	103	-293.0	-296.9 +/- 19.4	239	-227.0	-232.9 +/- 9.7	529	-501.0	-516.9 +/- 19.4
187	194	26	2.0	2.0 +/- 0.0	63	11.0	11.0 +/- 0.0	90	82.0	82.0 +/- 0.0
189	584	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0	2625	-2265.0	-1155.0 +/- 703.9
192	10	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0	1	-1.0	-1.0 +/- 0.0
193	0	8	-24.0	-24.0 +/- 0.0	12	-12.0	-12.0 +/- 0.0	23	-23.0	-23.0 +/- 0.0
194	8	0	0.0	0.0 +/- 0.0	1	-1.0	-1.0 +/- 0.0	19	-15.0	-15.0 +/- 0.0
195	0	14	-42.0	-42.0 +/- 0.0	18	-18.0	-18.0 +/- 0.0	59	-59.0	-59.0 +/- 0.0
202	583	27	-21.0	-21.0 +/- 0.0	142	-38.0	-37.4 +/- 29.0	818	122.0	217.4 +/- 405.7
207	1	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0
211	2	6	-14.0	-14.0 +/- 0.0	6	-4.0	-4.0 +/- 0.0	39	-31.0	-31.0 +/- 0.0
221	193	15	-33.0	-33.0 +/- 0.0	15	-9.0	-9.0 +/- 0.0	106	34.0	25.9 +/- 9.4
222	2	0	0.0	0.0 +/- 0.0	2	-2.0	-2.0 +/- 0.0	15	-15.0	-15.0 +/- 0.0
224	1	0	0.0	0.0 +/- 0.0	3	-3.0	-3.0 +/- 0.0	21	-21.0	-21.0 +/- 0.0
228	68	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0	12	8.0	8.0 +/- 0.0
240	88	0	0.0	0.0 +/- 0.0	0	0.0	0.0 +/- 0.0	16	8.0	8.0 +/- 0.0
243	2	0	0.0	0.0 +/- 0.0	4	-4.0	-4.0 +/- 0.0	20	-16.0	-16.0 +/- 0.0

Summary Statistics

Run Number iti96f—category A, automatic
Number of Topics 49

Total number of documents over all topics

	Run 1	Run 2	Run 3
Retrieved:	2801	5787	20184
Relevant:	5808	5808	5808
Rel_ret:	840	1340	2745

Average Effectiveness
(based on pool of submitted docs,
and averaged over 45 topics)

	Run 1	Run 2	Run 3
Mean Precision:	0.2222	0.1520	0.1417
Mean Recall:	0.1266	0.1993	0.3486

Table 1: Raw Data

topic	num rel	Run 1					Run 2					Run 3				
		set size	pool util.	sample utility			set size	pool util.	sample utility			set size	pool util.	sample utility		
1	30	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
3	101	43	-5.0	-7.6	+/-	12.2	79	19.0	16.6	+/-	7.1	158	130.0	100.3	+/-	37.0
4	178	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	15	41.0	41.0	+/-	0.0
5	19	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	39	1.0	1.0	+/-	0.0
6	158	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	57	87.0	87.0	+/-	0.0
11	92	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	7	1.0	1.0	+/-	0.0
12	228	7	-1.0	-1.0	+/-	0.0	29	17.0	17.0	+/-	0.0	109	171.0	167.8	+/-	12.4
23	7	0	0.0	0.0	+/-	0.0	1	1.0	1.0	+/-	0.0	1	3.0	3.0	+/-	0.0
24	38	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	13	7.0	7.0	+/-	0.0
44	10	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
53	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
54	48	4	0.0	0.0	+/-	0.0	10	4.0	4.0	+/-	0.0	56	16.0	16.0	+/-	0.0
58	45	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	7	21.0	21.0	+/-	0.0
68	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
77	14	1	-3.0	-3.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	15	1.0	1.0	+/-	0.0
78	37	32	-52.0	-52.0	+/-	0.0	114	-56.0	-72.7	+/-	13.8	148	-12.0	-45.4	+/-	27.6
82	55	10	2.0	2.0	+/-	0.0	35	9.0	9.0	+/-	0.0	51	49.0	49.0	+/-	0.0
94	56	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
95	93	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
100	157	21	21.0	21.0	+/-	0.0	92	62.0	62.0	+/-	0.0	92	216.0	216.0	+/-	0.0
108	174	1	1.0	1.0	+/-	0.0	70	-34.0	-34.0	+/-	0.0	80	12.0	12.0	+/-	0.0
111	887	386	-42.0	22.7	+/-	213.4	867	123.0	510.4	+/-	163.6	1092	1204.0	2372.0	+/-	347.8
114	42	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	2.0	2.0	+/-	0.0
118	324	0	0.0	0.0	+/-	0.0	1	1.0	1.0	+/-	0.0	9	27.0	27.0	+/-	0.0
119	185	2	-6.0	-6.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0	9	15.0	15.0	+/-	0.0
121	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
123	57	2	2.0	2.0	+/-	0.0	2	2.0	2.0	+/-	0.0	15	9.0	9.0	+/-	0.0
125	6	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
126	18	4	4.0	4.0	+/-	0.0	6	4.0	4.0	+/-	0.0	6	14.0	14.0	+/-	0.0
142	808	0	0.0	0.0	+/-	0.0	1	1.0	1.0	+/-	0.0	12	36.0	36.0	+/-	0.0
154	22	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	6.0	6.0	+/-	0.0
161	153	5	5.0	5.0	+/-	0.0	31	29.0	29.0	+/-	0.0	38	110.0	110.0	+/-	0.0
173	15	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	3.0	3.0	+/-	0.0
185	14	1	-3.0	-3.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	3	1.0	1.0	+/-	0.0
187	194	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	6.0	6.0	+/-	0.0
189	584	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	3.0	3.0	+/-	0.0
192	10	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
193	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
194	8	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
195	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
202	583	342	-202.0	-16.3	+/-	172.4	359	75.0	167.9	+/-	86.2	993	639.0	1359.6	+/-	417.2
207	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
211	2	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
221	193	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
222	2	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
224	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
228	68	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
240	88	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	5	3.0	3.0	+/-	0.0
243	2	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0

Summary Statistics

Run Number pircs96f—category A, automatic
Number of Topics 49

Total number of documents over all topics

	Run 1	Run 2	Run 3
Retrieved:	861	1703	3046
Relevant:	5808	5808	5808
Rel_ret:	576	977	1465

Average Effectiveness
(based on pool of submitted docs,
and averaged over 45 topics)

	Run 1	Run 2	Run 3
Mean Precision:	0.2145	0.2459	0.4237
Mean Recall:	0.0429	0.0910	0.1621

Table 1: Raw Data																
topic	num rel	Run 1					Run 2					Run 3				
		set size	pool util.	sample utility			set size	pool util.	sample utility			set size	pool util.	sample utility		
1	30	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
3	101	8	8.0	8.0	+/-	0.0	17	11.0	11.0	+/-	0.0	51	53.0	53.0	+/-	0.0
4	178	4	0.0	0.0	+/-	0.0	5	3.0	3.0	+/-	0.0	11	29.0	29.0	+/-	0.0
5	19	1	1.0	1.0	+/-	0.0	5	-1.0	-1.0	+/-	0.0	25	-9.0	-9.0	+/-	0.0
6	158	5	1.0	1.0	+/-	0.0	6	4.0	4.0	+/-	0.0	13	11.0	11.0	+/-	0.0
11	92	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	7	-7.0	-7.0	+/-	0.0
12	228	3	3.0	3.0	+/-	0.0	3	3.0	3.0	+/-	0.0	7	21.0	21.0	+/-	0.0
23	7	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
24	38	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	8	4.0	4.0	+/-	0.0
44	10	0	0.0	0.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0	6	-6.0	-6.0	+/-	0.0
53	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
54	48	20	-24.0	-24.0	+/-	0.0	36	-6.0	-6.0	+/-	0.0	68	4.0	4.0	+/-	0.0
58	45	0	0.0	0.0	+/-	0.0	2	0.0	0.0	+/-	0.0	39	-31.0	-31.0	+/-	0.0
68	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
77	14	1	-3.0	-3.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0	4	-4.0	-4.0	+/-	0.0
78	37	1	-3.0	-3.0	+/-	0.0	9	-1.0	-1.0	+/-	0.0	48	12.0	12.0	+/-	0.0
82	55	11	3.0	3.0	+/-	0.0	23	3.0	3.0	+/-	0.0	30	34.0	34.0	+/-	0.0
94	56	0	0.0	0.0	+/-	0.0	1	1.0	1.0	+/-	0.0	3	5.0	5.0	+/-	0.0
95	93	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
100	157	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	55	133.0	133.0	+/-	0.0
108	174	35	-41.0	-43.2	+/-	4.0	82	-32.0	-34.0	+/-	7.1	249	-13.0	69.6	+/-	98.8
111	887	473	-275.0	139.1	+/-	237.1	663	51.0	334.8	+/-	132.5	892	772.0	1464.5	+/-	301.3
114	42	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	7	-7.0	-7.0	+/-	0.0
118	324	8	8.0	8.0	+/-	0.0	23	21.0	21.0	+/-	0.0	65	115.0	115.0	+/-	0.0
119	185	4	0.0	0.0	+/-	0.0	17	-3.0	-3.0	+/-	0.0	75	37.0	37.0	+/-	0.0
121	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
123	57	0	0.0	0.0	+/-	0.0	2	0.0	0.0	+/-	0.0	14	18.0	18.0	+/-	0.0
125	6	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0
126	18	3	3.0	3.0	+/-	0.0	5	3.0	3.0	+/-	0.0	6	14.0	14.0	+/-	0.0
142	808	49	41.0	43.2	+/-	6.3	115	101.0	104.1	+/-	8.3	430	706.0	1153.7	+/-	119.9
154	22	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
161	153	0	0.0	0.0	+/-	0.0	10	10.0	10.0	+/-	0.0	43	125.0	125.0	+/-	0.0
173	15	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	2.0	2.0	+/-	0.0
185	14	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
187	194	0	0.0	0.0	+/-	0.0	1	1.0	1.0	+/-	0.0	10	18.0	18.0	+/-	0.0
189	584	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	26	42.0	42.0	+/-	0.0
192	10	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
193	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
194	8	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
195	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
202	583	247	-121.0	43.6	+/-	126.6	334	50.0	132.1	+/-	67.5	472	492.0	711.1	+/-	158.5
207	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
211	2	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
221	193	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
222	2	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
224	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
228	68	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	3	-3.0	-3.0	+/-	0.0
240	88	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	3	1.0	1.0	+/-	0.0
243	2	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0

Summary Statistics			
Run Number	xerox.fl—category A, automatic		
Number of Topics	49		
Total number of documents over all topics			
	Run 1	Run 2	Run 3
Retrieved:	873	1365	2678
Relevant:	5808	5808	5808
Rel_ret:	555	789	1310

Average Effectiveness (based on pool of submitted docs, and averaged over 45 topics)			
	Run 1	Run 2	Run 3
Mean Precision:	0.2493	0.3069	0.3024
Mean Recall:	0.0330	0.0541	0.1071

Table 1: Raw Data																
topic	num rel	Run 1					Run 2					Run 3				
		set size	pool util.	sample utility			set size	pool util.	sample utility			set size	pool util.	sample utility		
1	30	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
3	101	14	6.0	6.0	+/-	0.0	19	11.0	11.0	+/-	0.0	52	48.0	48.0	+/-	0.0
4	178	4	0.0	0.0	+/-	0.0	5	3.0	3.0	+/-	0.0	13	35.0	35.0	+/-	0.0
5	19	1	1.0	1.0	+/-	0.0	7	-1.0	-1.0	+/-	0.0	24	-4.0	-4.0	+/-	0.0
6	158	6	2.0	2.0	+/-	0.0	6	4.0	4.0	+/-	0.0	16	20.0	20.0	+/-	0.0
11	92	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	18	-10.0	-10.0	+/-	0.0
12	228	3	3.0	3.0	+/-	0.0	4	4.0	4.0	+/-	0.0	13	27.0	27.0	+/-	0.0
23	7	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
24	38	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	9	3.0	3.0	+/-	0.0
44	10	2	-6.0	-6.0	+/-	0.0	3	-3.0	-3.0	+/-	0.0	6	-6.0	-6.0	+/-	0.0
53	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
54	48	22	-26.0	-26.0	+/-	0.0	37	-5.0	-5.0	+/-	0.0	74	10.0	10.0	+/-	0.0
58	45	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	33	-25.0	-25.0	+/-	0.0
68	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
77	14	2	-6.0	-6.0	+/-	0.0	3	-3.0	-3.0	+/-	0.0	11	5.0	5.0	+/-	0.0
78	37	3	-5.0	-5.0	+/-	0.0	10	0.0	-0.0	+/-	0.0	23	17.0	17.0	+/-	0.0
82	55	16	-4.0	-4.0	+/-	0.0	28	6.0	6.0	+/-	0.0	37	35.0	35.0	+/-	0.0
94	56	0	0.0	0.0	+/-	0.0	1	1.0	1.0	+/-	0.0	5	3.0	3.0	+/-	0.0
95	93	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
100	157	0	0.0	0.0	+/-	0.0	15	11.0	11.0	+/-	0.0	58	138.0	138.0	+/-	0.0
108	174	42	-46.0	-46.9	+/-	12.5	99	-37.0	-38.7	+/-	11.7	263	-31.0	-23.2	+/-	81.8
111	887	532	-320.0	-31.3	+/-	309.9	706	62.0	255.6	+/-	164.2	896	808.0	1234.5	+/-	345.2
114	42	1	-3.0	-3.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0	14	-14.0	-14.0	+/-	0.0
118	324	12	12.0	12.0	+/-	0.0	28	16.0	16.0	+/-	0.0	83	121.0	121.0	+/-	0.0
119	185	4	0.0	0.0	+/-	0.0	22	-2.0	-2.0	+/-	0.0	73	39.0	39.0	+/-	0.0
121	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
123	57	2	-2.0	-2.0	+/-	0.0	2	0.0	0.0	+/-	0.0	10	22.0	22.0	+/-	0.0
125	6	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
126	18	4	0.0	0.0	+/-	0.0	6	4.0	4.0	+/-	0.0	7	13.0	13.0	+/-	0.0
142	808	51	43.0	39.0	+/-	9.5	112	102.0	102.3	+/-	6.8	346	686.0	848.4	+/-	116.7
154	22	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
161	153	3	3.0	3.0	+/-	0.0	17	17.0	17.0	+/-	0.0	44	128.0	128.0	+/-	0.0
173	15	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	2.0	2.0	+/-	0.0
185	14	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
187	194	0	0.0	0.0	+/-	0.0	2	2.0	2.0	+/-	0.0	8	16.0	16.0	+/-	0.0
189	584	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	33	67.0	67.0	+/-	0.0
192	10	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
193	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
194	8	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
195	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	3	-3.0	-3.0	+/-	0.0
202	583	264	-136.0	108.7	+/-	119.1	375	53.0	176.3	+/-	67.7	538	546.0	900.4	+/-	168.7
207	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
211	2	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
221	193	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	6.0	6.0	+/-	0.0
222	2	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
224	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
228	68	0	0.0	0.0	+/-	0.0	3	-3.0	-3.0	+/-	0.0	3	-3.0	-3.0	+/-	0.0
240	88	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	3	1.0	1.0	+/-	0.0
243	2	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	6	-2.0	-2.0	+/-	0.0

Summary Statistics			
Run Number	xerox.f2-category A, automatic		
Number of Topics	49		
Total number of documents over all topics			
	Run 1	Run 2	Run 3
Retrieved:	988	1517	2730
Relevant:	5808	5808	5808
Rel_ret:	620	876	1356

Average Effectiveness (based on pool of submitted docs, and averaged over 45 topics)			
	Run 1	Run 2	Run 3
Mean Precision:	0.2794	0.3159	0.3379
Mean Recall:	0.0387	0.0646	0.1280

Table 1: Raw Data

topic	num rel	Run 1					Run 2					Run 3				
		set size	pool util.	sample utility			set size	pool util.	sample utility			set size	pool util.	sample utility		
1	30	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
3	101	7	-1.0	-1.0	+/-	0.0	16	8.0	8.0	+/-	0.0	52	44.0	44.0	+/-	0.0
4	178	12	12.0	12.0	+/-	0.0	36	12.0	12.0	+/-	0.0	47	65.0	65.0	+/-	0.0
5	19	1	1.0	1.0	+/-	0.0	16	-8.0	-8.0	+/-	0.0	56	-12.0	-12.0	+/-	0.0
6	158	5	1.0	1.0	+/-	0.0	13	7.0	7.0	+/-	0.0	20	28.0	28.0	+/-	0.0
11	92	9	-23.0	-23.0	+/-	0.0	20	-14.0	-14.0	+/-	0.0	56	4.0	4.0	+/-	0.0
12	228	3	3.0	3.0	+/-	0.0	28	4.0	4.0	+/-	0.0	96	40.0	40.0	+/-	0.0
23	7	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	3	1.0	1.0	+/-	0.0
24	38	0	0.0	0.0	+/-	0.0	1	1.0	1.0	+/-	0.0	13	-5.0	-5.0	+/-	0.0
44	10	2	-6.0	-6.0	+/-	0.0	3	-3.0	-3.0	+/-	0.0	6	-6.0	-6.0	+/-	0.0
53	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
54	48	30	-38.0	-38.0	+/-	0.0	47	-15.0	-15.0	+/-	0.0	88	-4.0	-4.0	+/-	0.0
58	45	0	0.0	0.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0	83	-75.0	-75.0	+/-	0.0
68	0	1	-3.0	-3.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	22	-22.0	-22.0	+/-	0.0
77	14	14	-18.0	-18.0	+/-	0.0	15	-3.0	-3.0	+/-	0.0	19	5.0	5.0	+/-	0.0
78	37	24	-36.0	-36.0	+/-	0.0	53	-23.0	-23.0	+/-	0.0	53	7.0	7.0	+/-	0.0
82	55	18	-6.0	-6.0	+/-	0.0	30	4.0	4.0	+/-	0.0	39	33.0	33.0	+/-	0.0
94	56	4	-8.0	-8.0	+/-	0.0	5	-3.0	-3.0	+/-	0.0	9	-1.0	-1.0	+/-	0.0
95	93	2	-6.0	-6.0	+/-	0.0	19	-17.0	-17.0	+/-	0.0	28	-24.0	-24.0	+/-	0.0
100	157	15	7.0	7.0	+/-	0.0	47	31.0	31.0	+/-	0.0	69	135.0	135.0	+/-	0.0
108	174	63	-73.0	-63.0	+/-	29.2	137	-53.0	-47.1	+/-	20.7	296	-68.0	-39.1	+/-	76.3
111	887	491	-261.0	28.9	+/-	274.2	656	88.0	274.9	+/-	146.3	856	792.0	1296.8	+/-	316.9
114	42	2	-6.0	-6.0	+/-	0.0	12	-10.0	-10.0	+/-	0.0	57	-21.0	-21.0	+/-	0.0
118	324	18	2.0	2.0	+/-	0.0	51	21.0	21.0	+/-	0.0	127	125.0	153.5	+/-	25.5
119	185	10	-18.0	-18.0	+/-	0.0	45	-19.0	-19.0	+/-	0.0	286	-86.0	-58.7	+/-	87.1
121	0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
123	57	3	-1.0	-1.0	+/-	0.0	9	3.0	3.0	+/-	0.0	14	22.0	22.0	+/-	0.0
125	6	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	9	-5.0	-5.0	+/-	0.0
126	18	3	-1.0	-1.0	+/-	0.0	9	1.0	1.0	+/-	0.0	13	7.0	7.0	+/-	0.0
142	808	45	41.0	39.7	+/-	5.1	118	104.0	106.5	+/-	9.3	277	583.0	711.6	+/-	72.8
154	22	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
161	153	5	5.0	5.0	+/-	0.0	27	27.0	27.0	+/-	0.0	47	137.0	137.0	+/-	0.0
173	15	0	0.0	0.0	+/-	0.0	5	-3.0	-3.0	+/-	0.0	5	-1.0	-1.0	+/-	0.0
185	14	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	7	1.0	1.0	+/-	0.0
187	194	1	1.0	1.0	+/-	0.0	4	2.0	2.0	+/-	0.0	17	27.0	27.0	+/-	0.0
189	584	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	98	206.0	206.0	+/-	0.0
192	10	1	-3.0	-3.0	+/-	0.0	3	-3.0	-3.0	+/-	0.0	4	-4.0	-4.0	+/-	0.0
193	0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
194	8	1	-3.0	-3.0	+/-	0.0	5	-5.0	-5.0	+/-	0.0	12	-12.0	-12.0	+/-	0.0
195	0	1	-3.0	-3.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	77	-77.0	-77.0	+/-	0.0
202	583	270	-210.0	-111.2	+/-	164.6	420	24.0	129.4	+/-	93.0	611	493.0	811.9	+/-	220.8
207	1	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	16	-16.0	-16.0	+/-	0.0
211	2	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	3	-3.0	-3.0	+/-	0.0
221	193	0	0.0	0.0	+/-	0.0	2	0.0	0.0	+/-	0.0	5	7.0	7.0	+/-	0.0
222	2	1	-3.0	-3.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0
224	1	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
228	68	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
240	88	1	1.0	1.0	+/-	0.0	18	-8.0	-8.0	+/-	0.0	30	6.0	6.0	+/-	0.0
243	2	2	-6.0	-6.0	+/-	0.0	5	-3.0	-3.0	+/-	0.0	7	-3.0	-3.0	+/-	0.0

Summary Statistics

Run Number xerox.f3-category A, automatic
Number of Topics 49

Total number of documents over all topics

	Run 1	Run 2	Run 3
Retrieved:	1065	1884	3636
Relevant:	5808	5808	5808
Rel_ret:	634	1011	1489

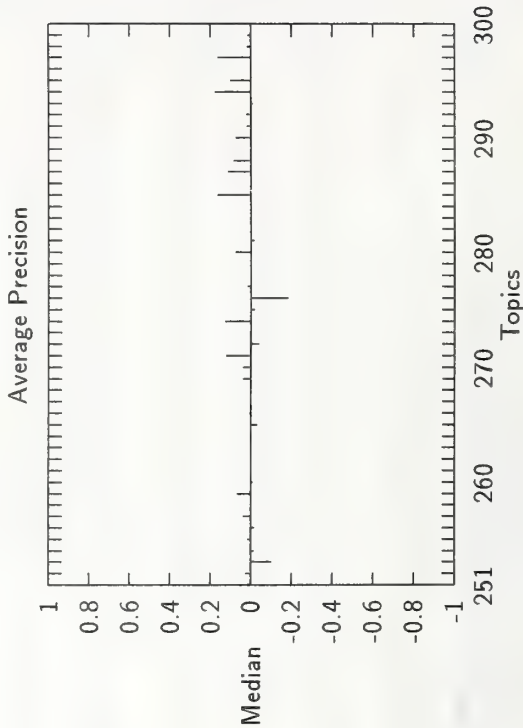
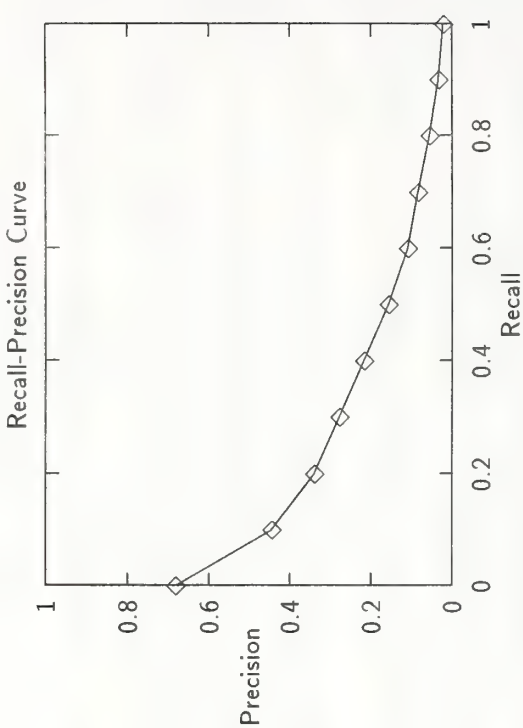
Average Effectiveness
(based on pool of submitted docs,
and averaged over 45 topics)

	Run 1	Run 2	Run 3
Mean Precision:	0.3482	0.3200	0.2907
Mean Recall:	0.0568	0.1111	0.1711

Summary Statistics	
Run Number	anu5mrg0-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	29856
Relevant:	5524
Rel_ret:	2564

Recall Level Precision Averages	
Recall	Precision
0.00	0.6815
0.10	0.4458
0.20	0.3400
0.30	0.2786
0.40	0.2175
0.50	0.1572
0.60	0.1101
0.70	0.0842
0.80	0.0564
0.90	0.0328
1.00	0.0203
Average precision over all relevant docs	
non-interpolated	0.1973

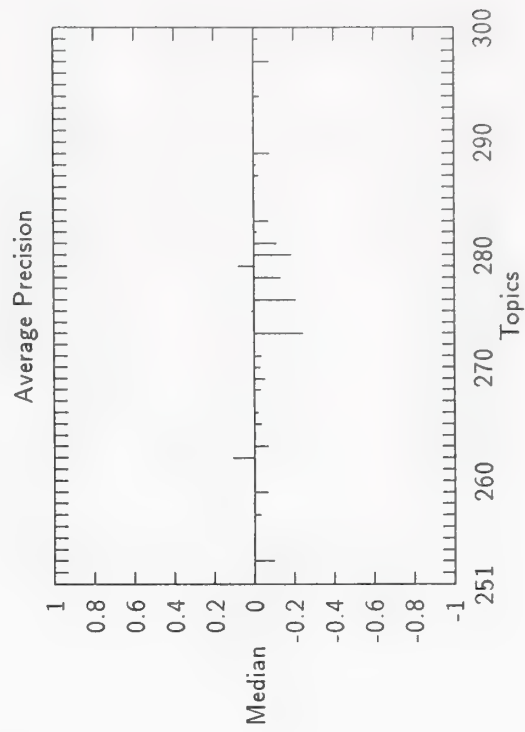
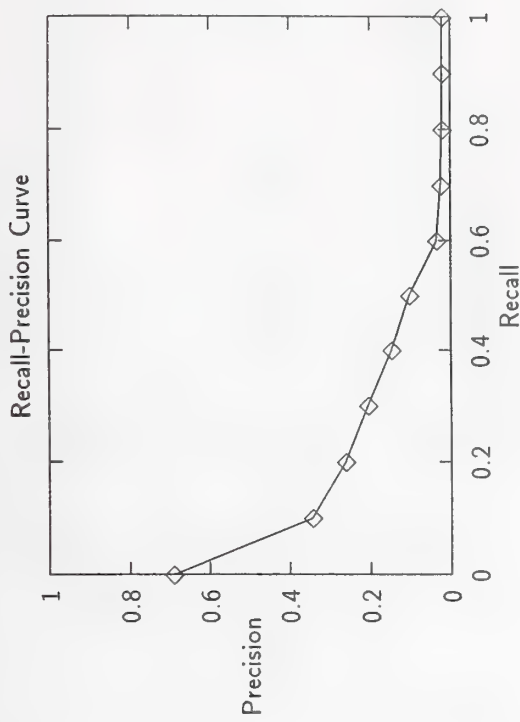
Document Level Averages	
At 5 docs	0.4400
At 10 docs	0.4020
At 15 docs	0.3787
At 20 docs	0.3650
At 30 docs	0.3313
At 100 docs	0.2238
At 200 docs	0.1582
At 500 docs	0.0860
At 1000 docs	0.0513
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2395



Summary Statistics	
Run Number	anu5mrg1—category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	25340
Relevant:	5524
Rel_ret:	1612

Recall Level Precision Averages	
Recall	Precision
0.00	0.6905
0.10	0.3442
0.20	0.2627
0.30	0.2073
0.40	0.1471
0.50	0.1027
0.60	0.0350
0.70	0.0238
0.80	0.0200
0.90	0.0200
1.00	0.0200
Average precision over all relevant docs	
non-interpolated	0.1441

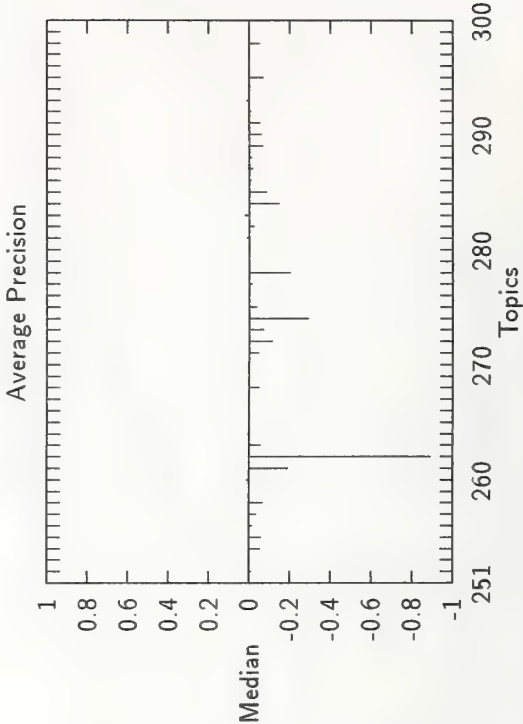
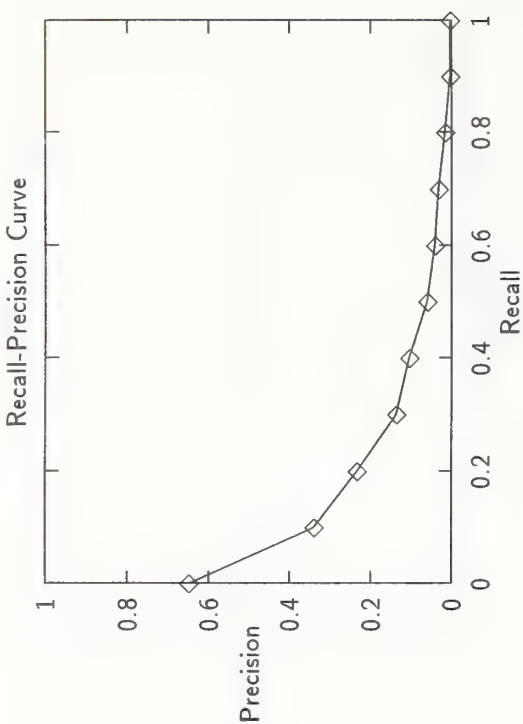
Document Level Averages	
At 5 docs	0.4440
At 10 docs	0.3740
At 15 docs	0.3493
At 20 docs	0.3140
At 30 docs	0.2827
At 100 docs	0.1712
At 200 docs	0.1061
At 500 docs	0.0550
At 1000 docs	0.0322
R—Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2091



Summary Statistics	
Run Number	anu5mrg7-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	24985
Relevant:	5524
Rel_ret:	1511

Recall Level Precision Averages	
Recall	Precision
0.00	0.6491
0.10	0.3395
0.20	0.2340
0.30	0.1370
0.40	0.1042
0.50	0.0600
0.60	0.0417
0.70	0.0317
0.80	0.0165
0.90	0.0025
1.00	0.0025
Average precision over all relevant docs	
non-interpolated	0.1200

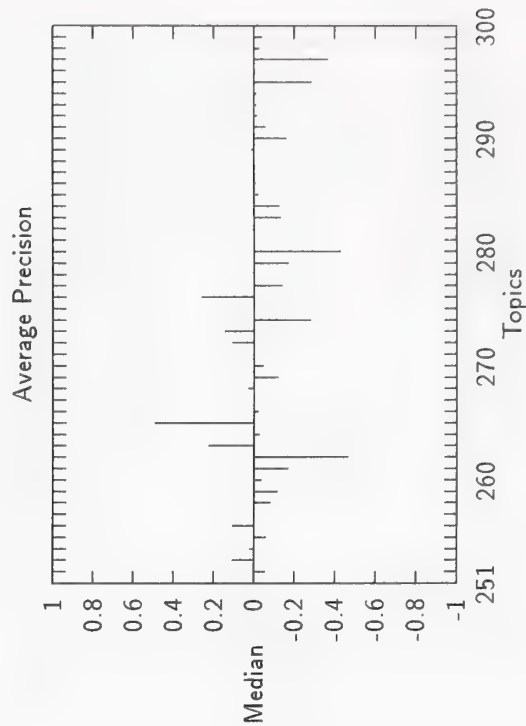
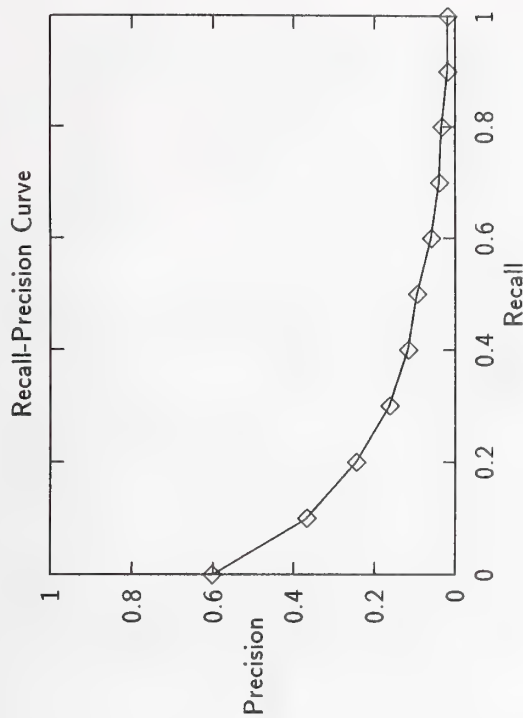
Document Level Averages	
At 5 docs	0.4200
At 10 docs	0.3720
At 15 docs	0.3480
At 20 docs	0.3150
At 30 docs	0.2747
At 100 docs	0.1666
At 200 docs	0.1096
At 500 docs	0.0536
At 1000 docs	0.0302
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1724



Summary Statistics	
Run Number	fsclt3m-category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	31189
Relevant:	5524
Rel_ret:	1389

Recall Level Precision Averages	
Recall	Precision
0.00	0.6019
0.10	0.3694
0.20	0.2458
0.30	0.1640
0.40	0.1182
0.50	0.0960
0.60	0.0612
0.70	0.0396
0.80	0.0337
0.90	0.0197
1.00	0.0197
Average precision over all relevant docs	
non-interpolated	0.1354

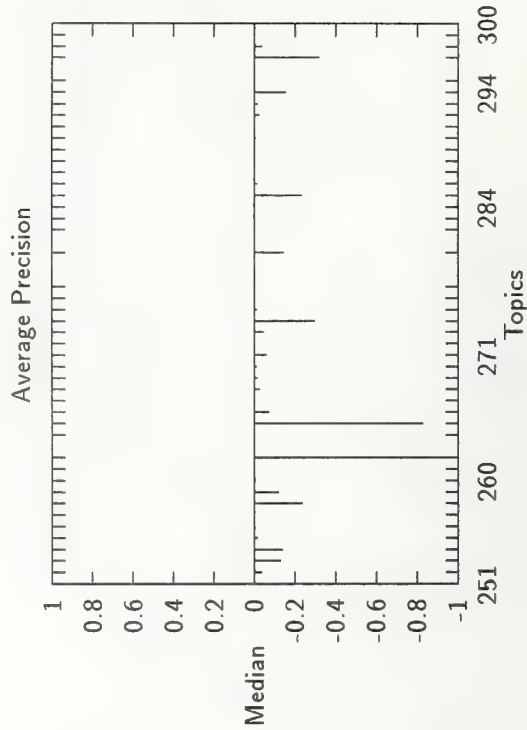
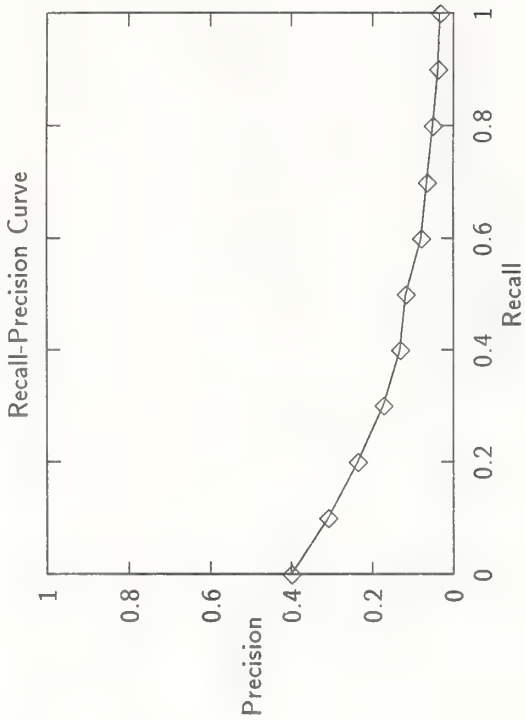
Document Level Averages	
	Precision
At 5 docs	0.3720
At 10 docs	0.3240
At 15 docs	0.3013
At 20 docs	0.2760
At 30 docs	0.2467
At 100 docs	0.1606
At 200 docs	0.1063
At 500 docs	0.0531
At 1000 docs	0.0278
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1890



Summary Statistics	
Run Number	UniNEO-category B, automatic
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	589

Recall Level Precision Averages	
Recall	Precision
0.00	0.4001
0.10	0.3100
0.20	0.2374
0.30	0.1735
0.40	0.1326
0.50	0.1179
0.60	0.0800
0.70	0.0649
0.80	0.0503
0.90	0.0389
1.00	0.0337
Average precision over all relevant docs	
non-interpolated	0.1335

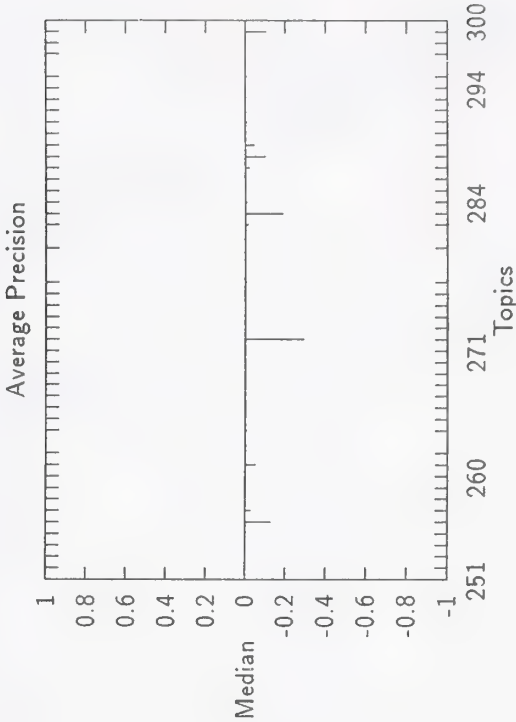
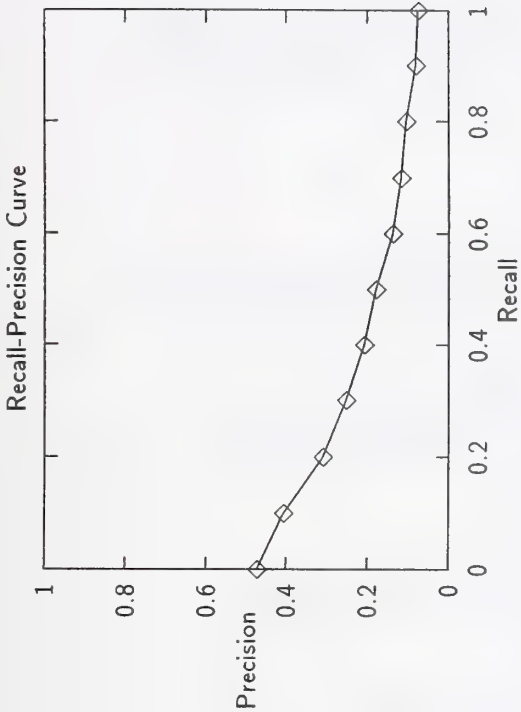
Document Level Averages	
At 5 docs	0.2356
At 10 docs	0.1578
At 15 docs	0.1407
At 20 docs	0.1189
At 30 docs	0.1037
At 100 docs	0.0560
At 200 docs	0.0390
At 500 docs	0.0214
At 1000 docs	0.0131
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1602

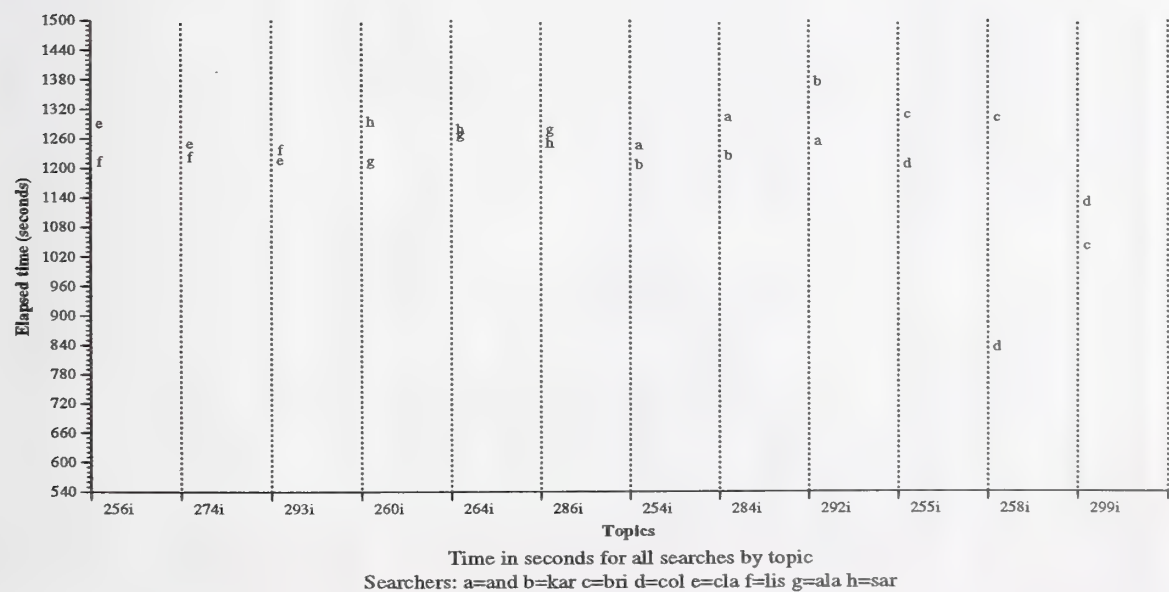
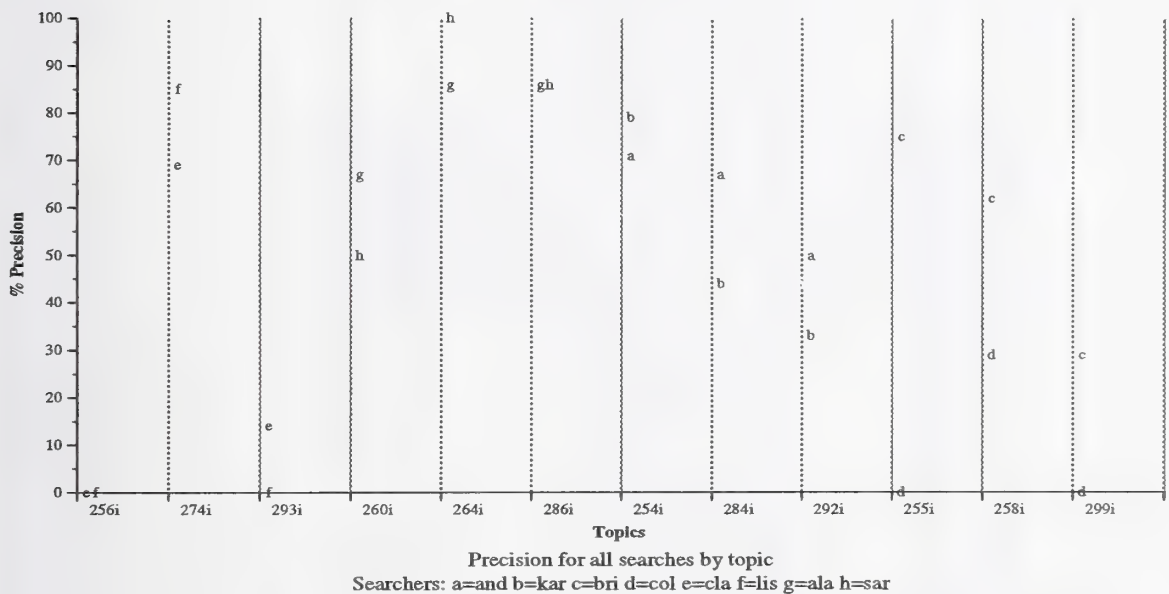
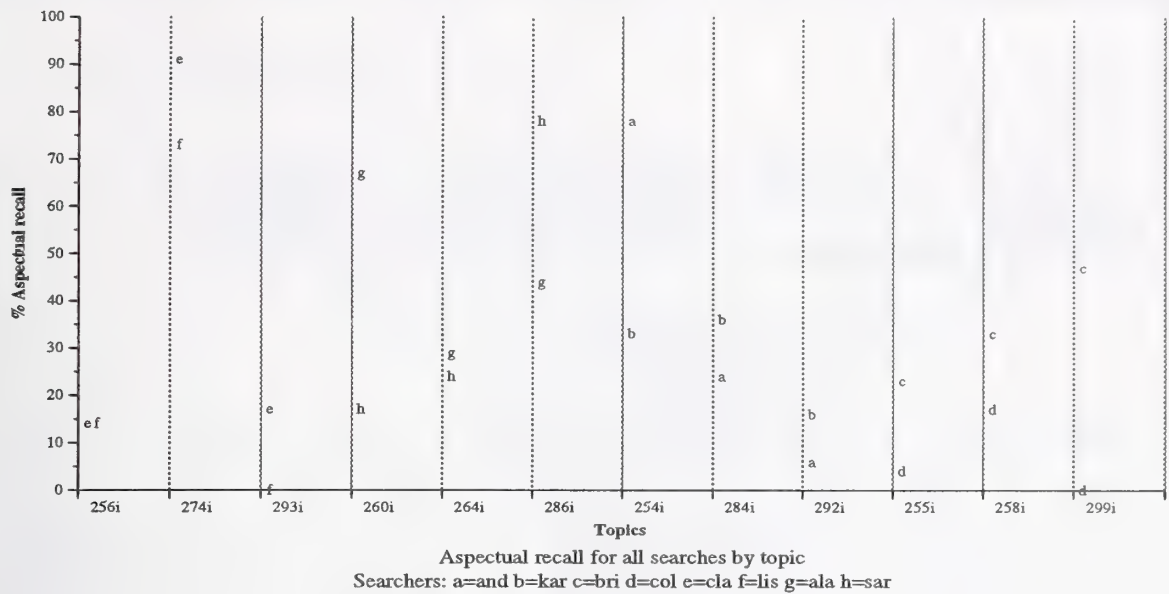


Summary Statistics	
Run Number	UniNE9-category B, automatic
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	654

Recall Level Precision Averages	
Recall	Precision
0.00	0.4717
0.10	0.4064
0.20	0.3100
0.30	0.2522
0.40	0.2083
0.50	0.1802
0.60	0.1388
0.70	0.1177
0.80	0.1056
0.90	0.0806
1.00	0.0752
Average precision over all relevant docs	
non-interpolated	0.1985

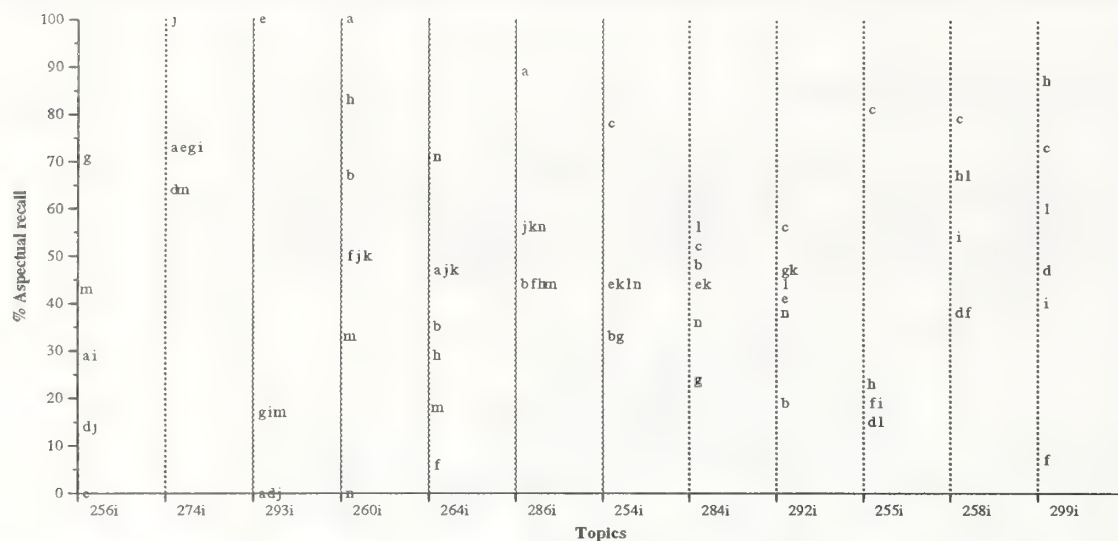
Document Level Averages	
	Precision
At 5 docs	0.2400
At 10 docs	0.1911
At 15 docs	0.1630
At 20 docs	0.1522
At 30 docs	0.1304
At 100 docs	0.0676
At 200 docs	0.0447
At 500 docs	0.0243
At 1000 docs	0.0145
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2061





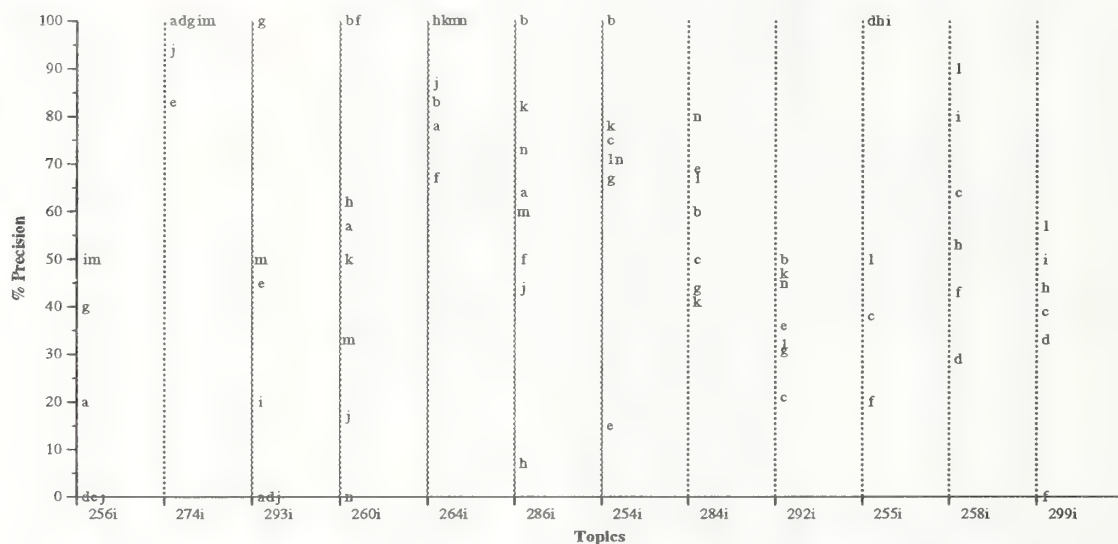
Summary Results for Each Search (continued)

Search id	Searcher id	Block number	Block sequence number	Topic	Elapsed time (secs)	Num docs saved	Precision(%)	Aspectual Recall(%)	Summary vector of aspects covered (1=covered; leftmost = #1)
274i-001-1	a	B1	1	274	1095	10	100	73	11111110100
274i-004-2	d	B1	2	274	1047	8	100	64	11111100100
274i-005-1	e	B1	1	274	1385	6	83	73	01111110110
274i-007-2	g	B1	2	274	712	14	100	73	11111110100
274i-009-1	i	B1	1	274	1247	5	100	73	11111110100
274i-010-2	j	B1	2	274	1243	36	94	100	11111111111
274i-013-1	m	B1	1	274	1191	6	100	64	11111100010
284i-002-2	b	B3	2	284	1157	15	60	48	1100011000010110001111001
284i-003-1	c	B3	1	284	1188	22	50	52	1101101001010110001110100
284i-005-2	e	B3	2	284	1157	13	69	44	1101111000000100011101000
284i-007-1	g	B3	1	284	1015	9	44	24	1000100000000110010010000
284i-011-1	k	B3	1	284	1018	17	41	44	1101101000011101011000000
284i-012-2	l	B3	2	284	1201	18	67	56	0111101110010111101000010
284i-014-2	n	B3	2	284	818	10	80	36	0001101000010111011000000
286i-001-2	a	B2	2	286	1152	28	64	89	110111111
286i-002-1	b	B2	1	286	939	6	100	44	110100001
286i-006-1	f	B2	1	286	1226	4	50	44	011110000
286i-008-2	h	B2	2	286	1121	14	7	44	110110000
286i-010-1	j	B2	1	286	1013	9	44	56	111100010
286i-011-2	k	B2	2	286	1251	11	82	56	110110001
286i-013-2	m	B2	2	286	590	5	60	44	110101000
286i-014-1	n	B2	1	286	1087	15	73	56	110110001
292i-002-2	b	B3	2	292	1110	4	50	19	00000001000110000000010000110000
292i-003-1	c	B3	1	292	1067	19	21	56	10111110110111000000101111001100
292i-005-2	e	B3	2	292	1209	22	36	41	00010001001100110011010000111001
292i-007-1	g	B3	1	292	1254	16	31	47	00010001001100111111110000111000
292i-011-1	k	B3	1	292	1213	19	47	47	00010001001100110111110000111010
292i-012-2	l	B3	2	292	1212	22	32	44	10010001001100110011110000111000
292i-014-2	n	B3	2	292	945	11	45	38	00010001001100110011010000111000
293i-001-1	a	B1	1	293	1206	5	0	0	000000
293i-004-2	d	B1	2	293	1357	0	0	0	
293i-005-1	e	B1	1	293	1261	11	45	100	111111
293i-007-2	g	B1	2	293	1337	1	100	17	000100
293i-009-1	i	B1	1	293	1136	5	20	17	000100
293i-010-2	j	B1	2	293	964	6	0	0	000000
293i-013-1	m	B1	1	293	1088	2	50	17	100000
299i-003-2	c	B4	2	299	1195	18	39	73	111111111100010
299i-004-1	d	B4	1	299	1251	3	33	47	101111101000000
299i-006-2	f	B4	2	299	1337	6	0	7	000000000000100
299i-008-1	h	B4	1	299	1140	9	44	87	10111111111011
299i-009-2	i	B4	2	299	1236	2	50	40	101111100000000
299i-012-1	l	B4	1	299	1157	7	57	60	111111110000010



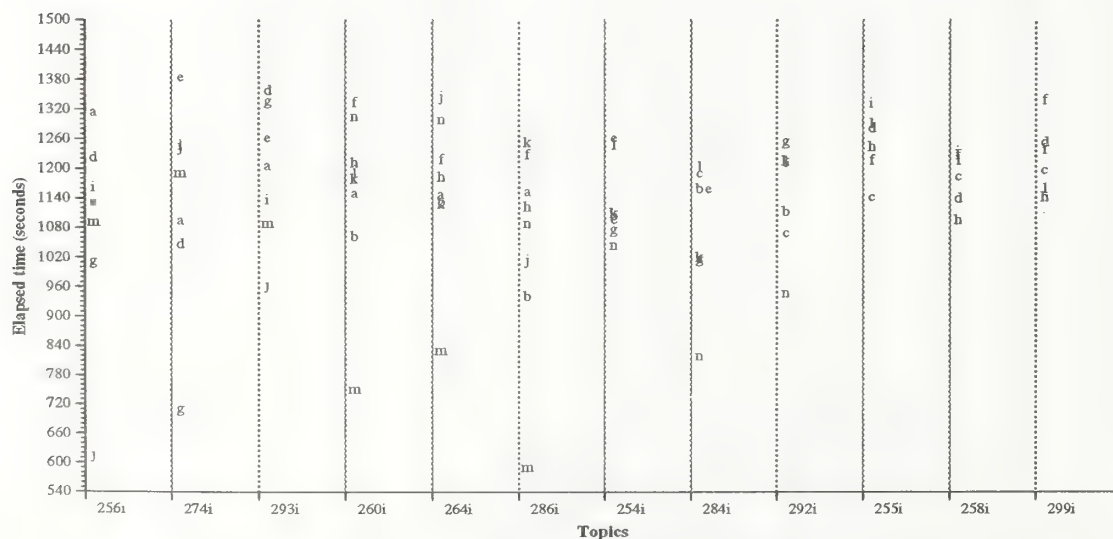
Aspectual recall of all searches by topic

Searchers: a=s001 b=s002 c=s003 d=s004 e=s005 f=s006 g=s007 h=s008 i=s009 j=s010 k=s011 l=s012 m=s013 n=s014



Precision of all searches by topic

Searchers: a=s001 b=s002 c=s003 d=s004 e=s005 f=s006 g=s007 h=s008 i=s009 j=s010 k=s011 l=s012 m=s013 n=s014



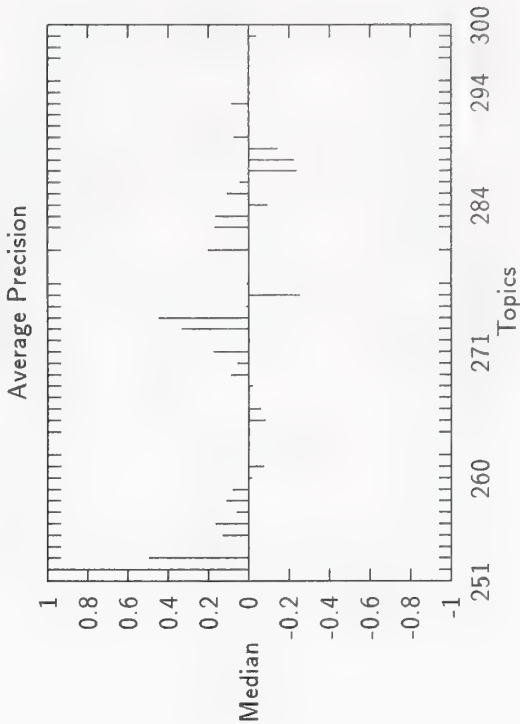
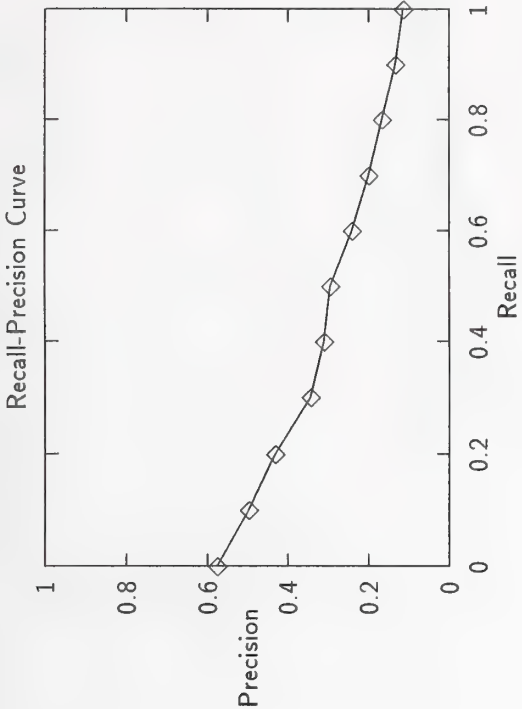
Time in seconds for all searches by topic

Searchers: a=s001 b=s002 c=s003 d=s004 e=s005 f=s006 g=s007 h=s008 i=s009 j=s010 k=s011 l=s012 m=s013 n=s014

Summary Statistics	
Run Number	CLATMC-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel.Ret:	766

Recall Level Precision Averages	
Recall	Precision
0.00	0.5752
0.10	0.4978
0.20	0.4332
0.30	0.3447
0.40	0.3111
0.50	0.2950
0.60	0.2421
0.70	0.2001
0.80	0.1676
0.90	0.1343
1.00	0.1144
Average precision over all relevant docs	
non-interpolated	0.2842

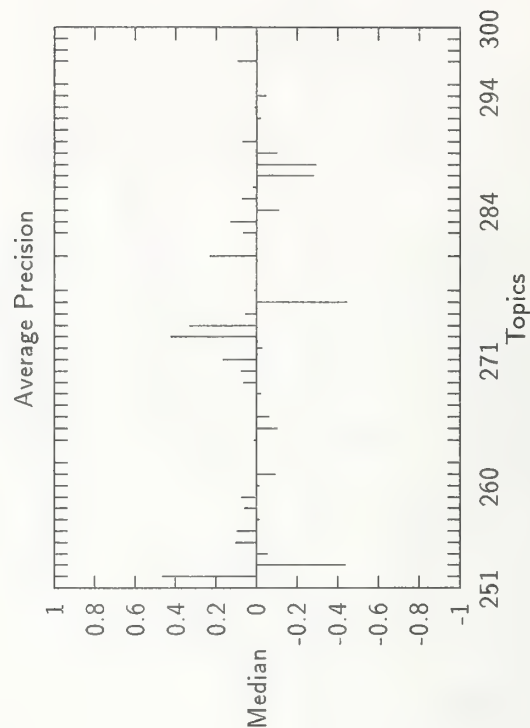
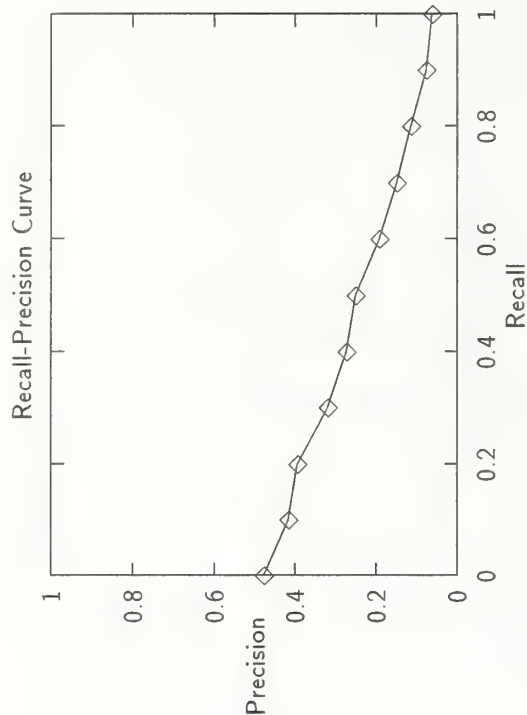
Document Level Averages	
	Precision
At 5 docs	0.3333
At 10 docs	0.2778
At 15 docs	0.2370
At 20 docs	0.2122
At 30 docs	0.1793
At 100 docs	0.0984
At 200 docs	0.0618
At 500 docs	0.0304
At 1000 docs	0.0170
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2934



Summary Statistics	
Run Number	CLATMN-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	786

Recall Level Precision Averages	
Recall	Precision
0.00	0.4767
0.10	0.4171
0.20	0.3939
0.30	0.3197
0.40	0.2732
0.50	0.2509
0.60	0.1917
0.70	0.1500
0.80	0.1150
0.90	0.0758
1.00	0.0614
Average precision over all relevant docs	
non-interpolated	0.2346

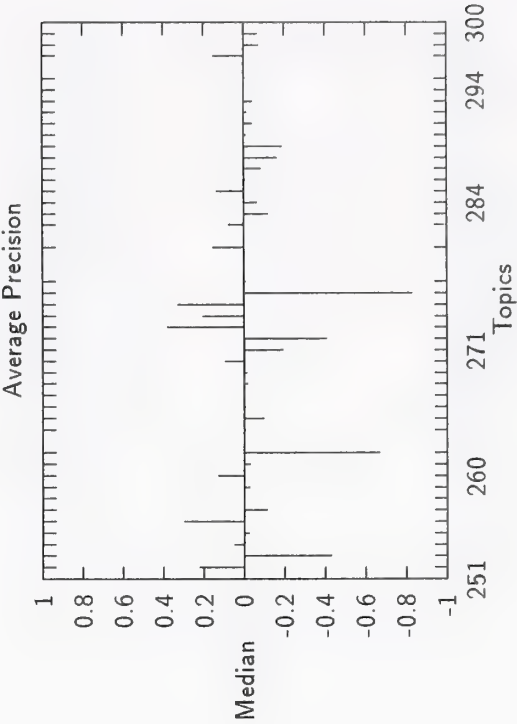
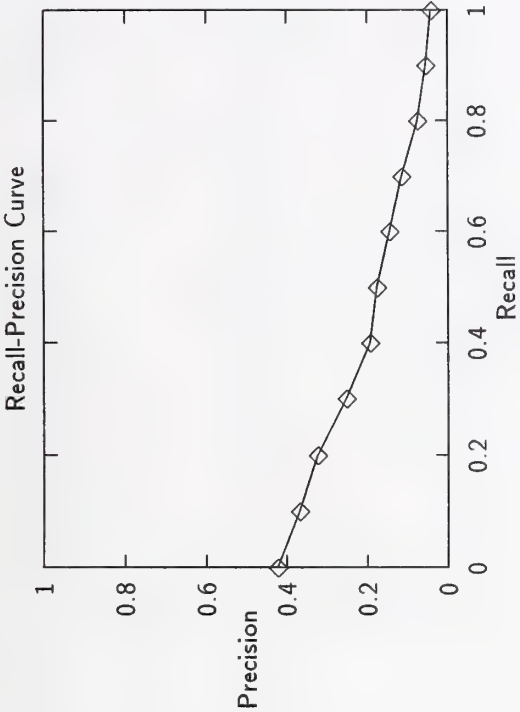
Document Level Averages	
	Precision
At 5 docs	0.3022
At 10 docs	0.2533
At 15 docs	0.2252
At 20 docs	0.2044
At 30 docs	0.1778
At 100 docs	0.0976
At 200 docs	0.0603
At 500 docs	0.0308
At 1000 docs	0.0175
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2345



Summary Statistics	
Run Number	CLPHR0-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	567

Recall Level Precision Averages	
Recall	Precision
0.00	0.4235
0.10	0.3686
0.20	0.3236
0.30	0.2528
0.40	0.1940
0.50	0.1766
0.60	0.1444
0.70	0.1144
0.80	0.0748
0.90	0.0534
1.00	0.0413
Average precision over all relevant docs	
non-interpolated	0.1834

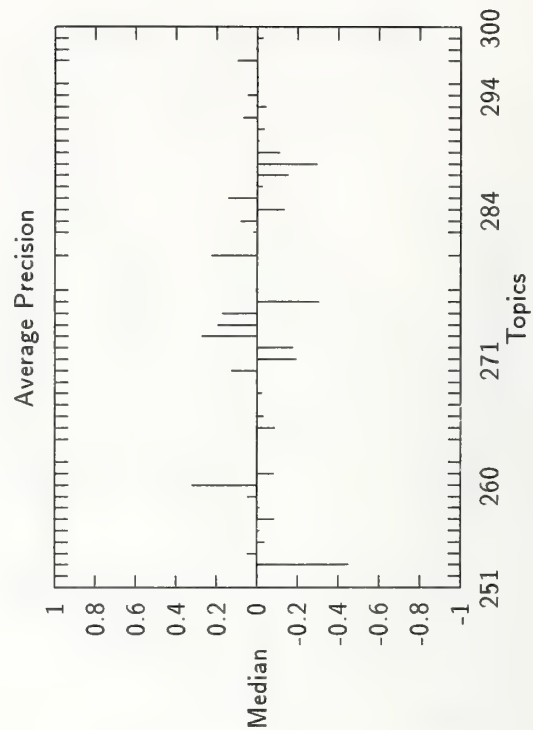
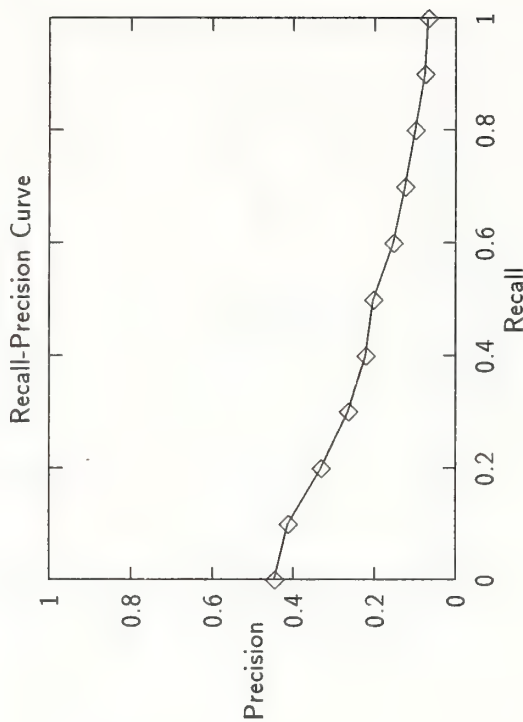
Document Level Averages	
At 5 docs	0.2578
At 10 docs	0.2156
At 15 docs	0.1733
At 20 docs	0.1567
At 30 docs	0.1281
At 100 docs	0.0667
At 200 docs	0.0431
At 500 docs	0.0224
At 1000 docs	0.0126
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1821



Summary Statistics	
Run Number	CLPHR1-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	609

Recall Level Precision Averages	
Recall	Precision
0.00	0.4465
0.10	0.4135
0.20	0.3320
0.30	0.2651
0.40	0.2223
0.50	0.2028
0.60	0.1540
0.70	0.1257
0.80	0.1004
0.90	0.0756
1.00	0.0688
Average precision over all relevant docs	
non-interpolated	0.2010

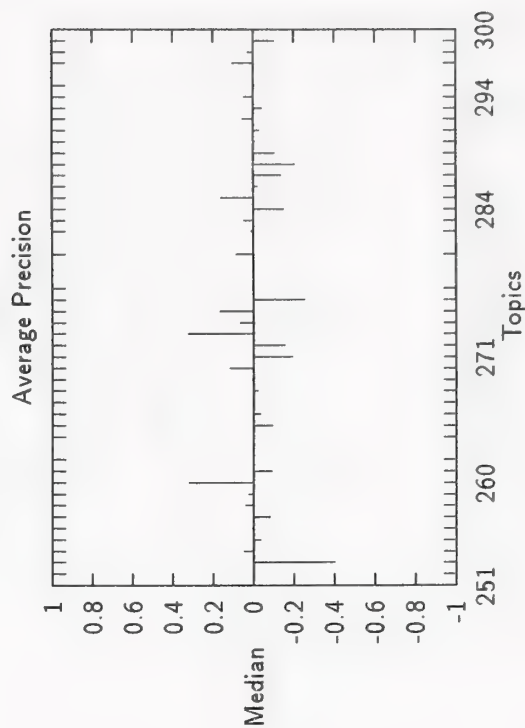
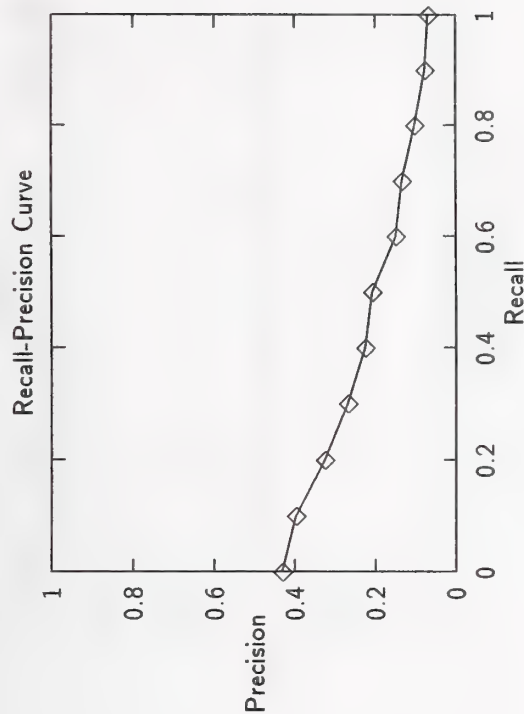
Document Level Averages	
	Precision
At 5 docs	0.2444
At 10 docs	0.2022
At 15 docs	0.1837
At 20 docs	0.1744
At 30 docs	0.1430
At 100 docs	0.0713
At 200 docs	0.0451
At 500 docs	0.0231
At 1000 docs	0.0135
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2127



Summary Statistics	
Run Number	CLPHR2-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
RelRet:	622

Recall Level Precision Averages	
Recall	Precision
0.00	0.4316
0.10	0.3973
0.20	0.3267
0.30	0.2684
0.40	0.2255
0.50	0.2078
0.60	0.1502
0.70	0.1330
0.80	0.1021
0.90	0.0760
1.00	0.0683
Average precision over all relevant docs	
non-interpolated	0.1997

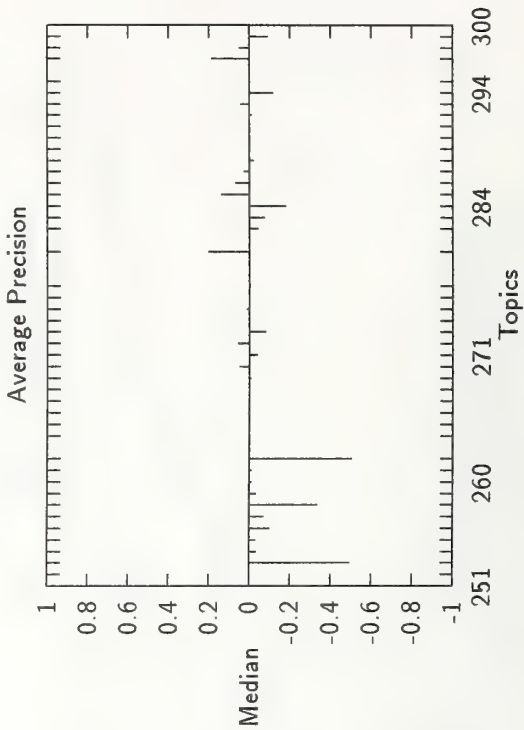
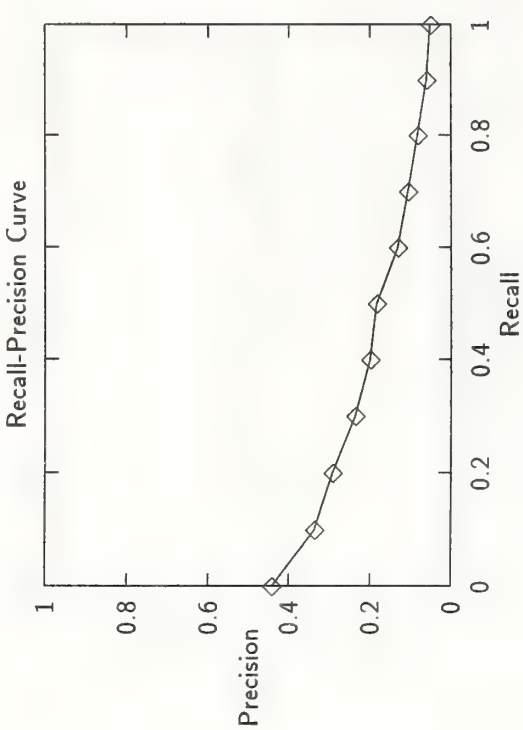
Document Level Averages	
	Precision
At 5 docs	0.2578
At 10 docs	0.2133
At 15 docs	0.1763
At 20 docs	0.1644
At 30 docs	0.1422
At 100 docs	0.0722
At 200 docs	0.0451
At 500 docs	0.0234
At 1000 docs	0.0138
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2044



Summary Statistics	
Run Number	genlp1-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	656

Recall Level Precision Averages	
Recall	Precision
0.00	0.4418
0.10	0.3369
0.20	0.2920
0.30	0.2366
0.40	0.1999
0.50	0.1840
0.60	0.1321
0.70	0.1072
0.80	0.0825
0.90	0.0607
1.00	0.0511
Average precision over all relevant docs	
non-interpolated	0.1773

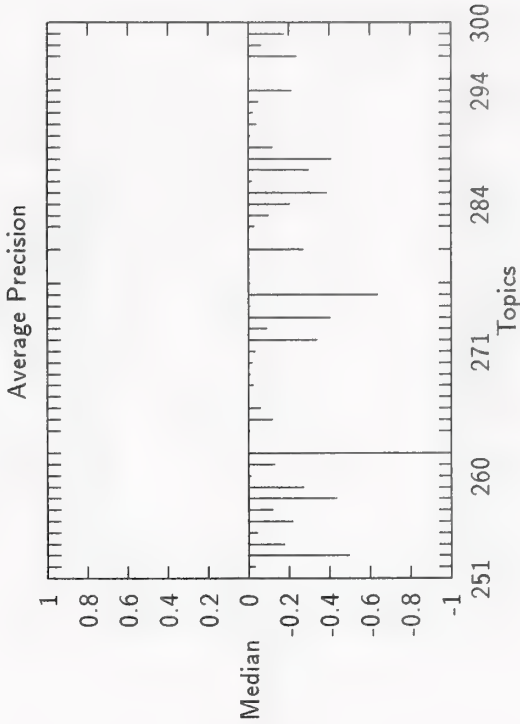
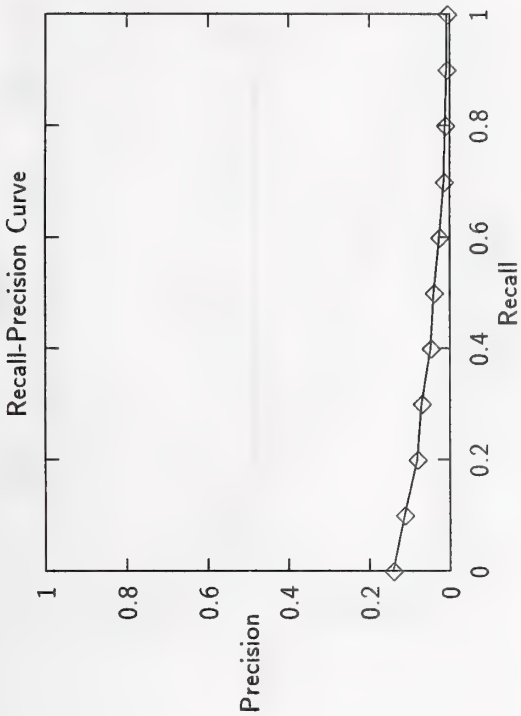
Document Level Averages	
At 5 docs	0.2311
At 10 docs	0.2044
At 15 docs	0.1630
At 20 docs	0.1456
At 30 docs	0.1230
At 100 docs	0.0664
At 200 docs	0.0438
At 500 docs	0.0246
At 1000 docs	0.0146
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1776



Summary Statistics	
Run Number	genlp2-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	444

Recall Level Precision Averages	
Recall	Precision
0.00	0.1415
0.10	0.1133
0.20	0.0818
0.30	0.0695
0.40	0.0474
0.50	0.0397
0.60	0.0251
0.70	0.0145
0.80	0.0106
0.90	0.0083
1.00	0.0062
Average precision over all relevant docs	
non-interpolated	0.0440

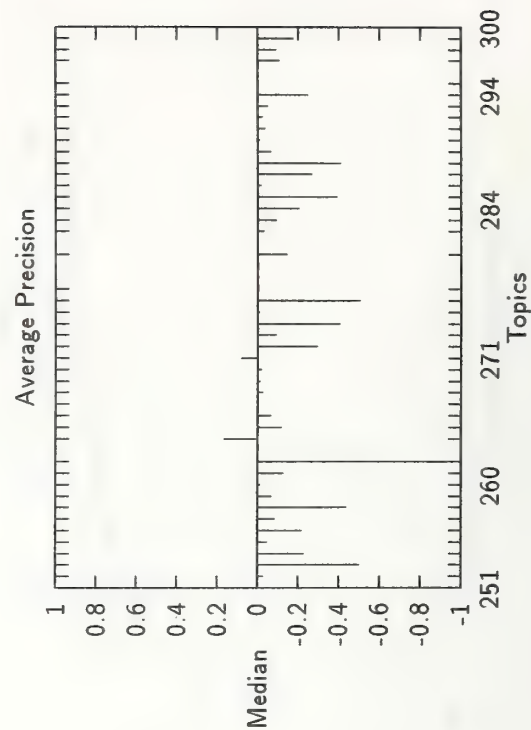
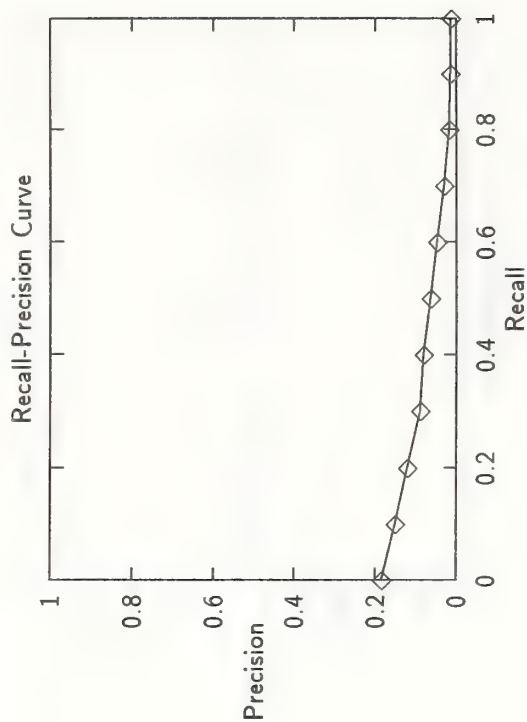
Document Level Averages	
	Precision
At 5 docs	0.0533
At 10 docs	0.0578
At 15 docs	0.0519
At 20 docs	0.0467
At 30 docs	0.0430
At 100 docs	0.0309
At 200 docs	0.0239
At 500 docs	0.0141
At 1000 docs	0.0099
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0583



Summary Statistics	
Run Number	genlp3-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	456

Recall Level Precision Averages	
Recall	Precision
0.00	0.1845
0.10	0.1507
0.20	0.1205
0.30	0.0897
0.40	0.0810
0.50	0.0641
0.60	0.0470
0.70	0.0303
0.80	0.0168
0.90	0.0136
1.00	0.0133
Average precision over all relevant docs	
non-interpolated	0.0654

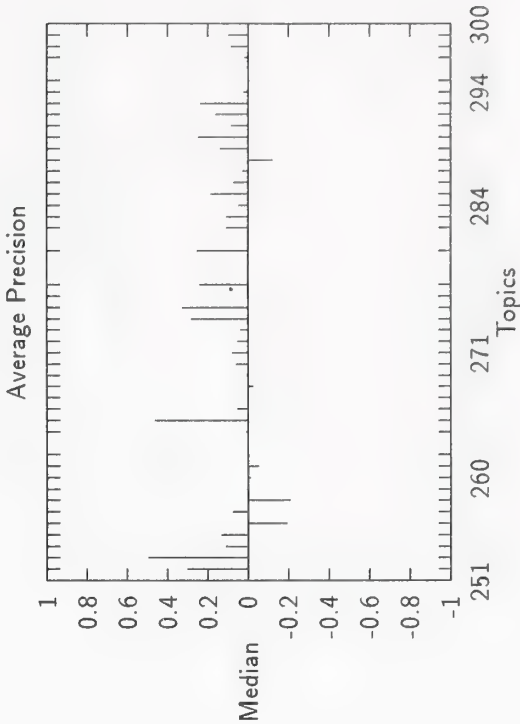
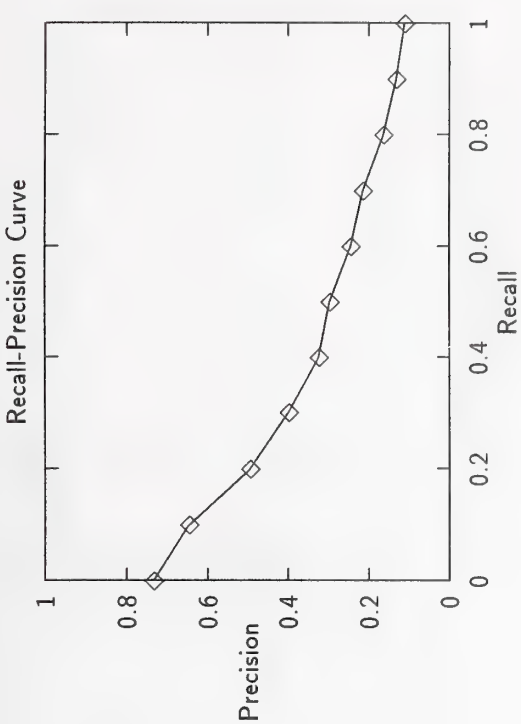
Document Level Averages	
	Precision
At 5 docs	0.0800
At 10 docs	0.0844
At 15 docs	0.0696
At 20 docs	0.0644
At 30 docs	0.0570
At 100 docs	0.0358
At 200 docs	0.0249
At 500 docs	0.0143
At 1000 docs	0.0101
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0818



Summary Statistics	
Run Number	genlp4-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	814

Recall Level Precision Averages	
Recall	Precision
0.00	0.7327
0.10	0.6462
0.20	0.4946
0.30	0.3987
0.40	0.3250
0.50	0.2974
0.60	0.2455
0.70	0.2144
0.80	0.1641
0.90	0.1327
1.00	0.1118
Average precision over all relevant docs	
non-interpolated	0.3176

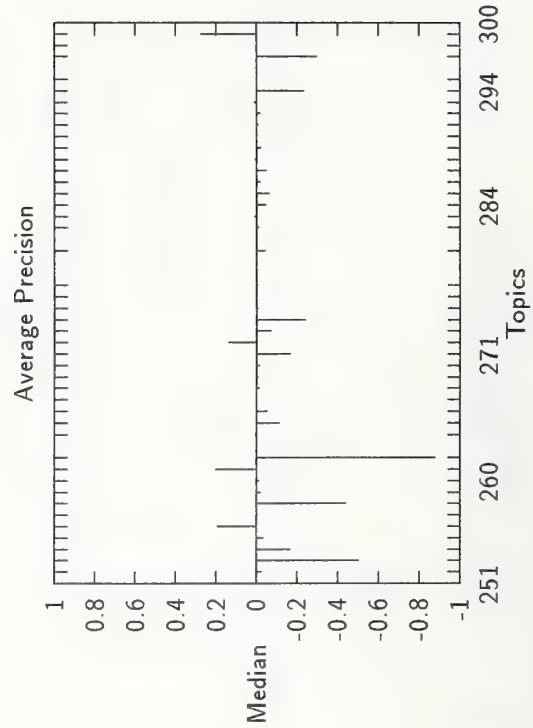
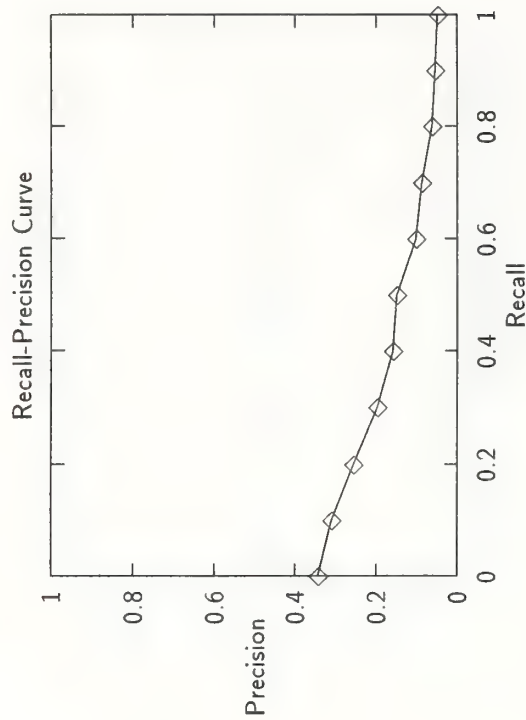
Document Level Averages	
	Precision
At 5 docs	0.3867
At 10 docs	0.3156
At 15 docs	0.2741
At 20 docs	0.2433
At 30 docs	0.2030
At 100 docs	0.0998
At 200 docs	0.0639
At 500 docs	0.0315
At 1000 docs	0.0181
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3091



Summary Statistics	
Run Number	sbase1.new-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	623

Recall Level Precision Averages	
Recall	Precision
0.00	0.3437
0.10	0.3098
0.20	0.2558
0.30	0.1964
0.40	0.1593
0.50	0.1498
0.60	0.1016
0.70	0.0882
0.80	0.0617
0.90	0.0543
1.00	0.0469
Average precision over all relevant docs	
non-interpolated	0.1478

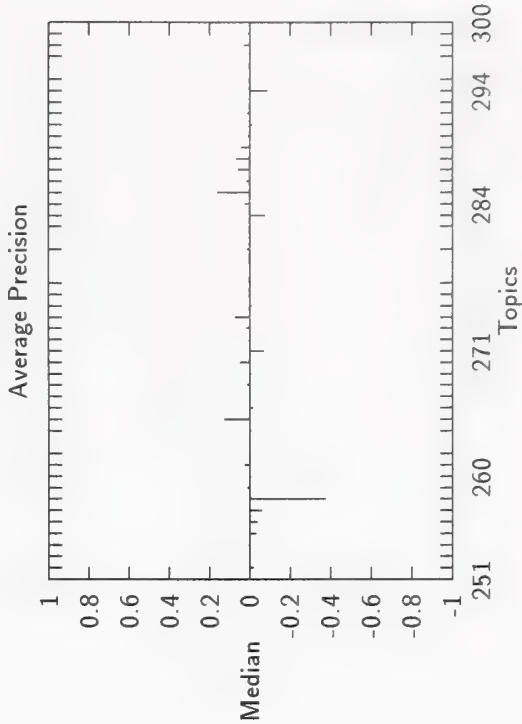
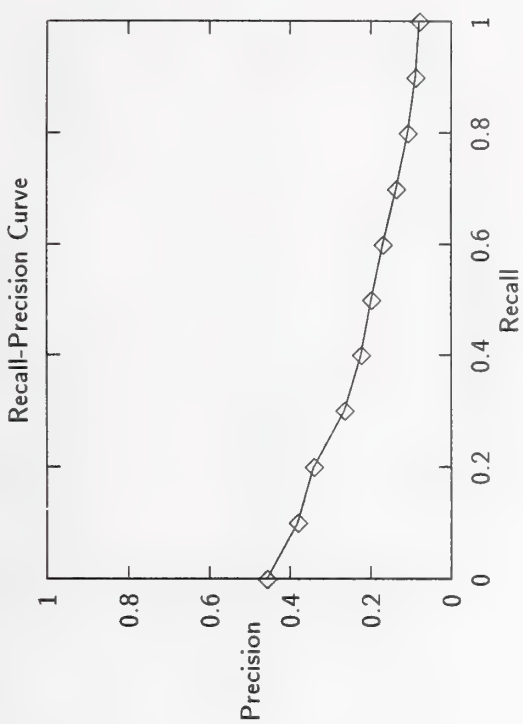
Document Level Averages	
	Precision
At 5 docs	0.1911
At 10 docs	0.1578
At 15 docs	0.1333
At 20 docs	0.1256
At 30 docs	0.1030
At 100 docs	0.0544
At 200 docs	0.0374
At 500 docs	0.0220
At 1000 docs	0.0138
R—Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1609



Summary Statistics	
Run Number	sbase2.new-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	689

Recall Level Precision Averages	
Recall	Precision
0.00	0.4578
0.10	0.3808
0.20	0.3413
0.30	0.2658
0.40	0.2248
0.50	0.1992
0.60	0.1707
0.70	0.1364
0.80	0.1082
0.90	0.0892
1.00	0.0796
Average precision over all relevant docs	
non-interpolated	0.2078

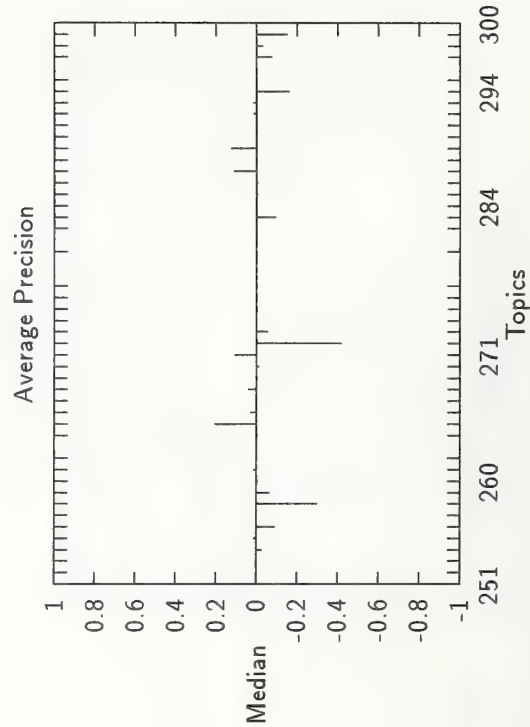
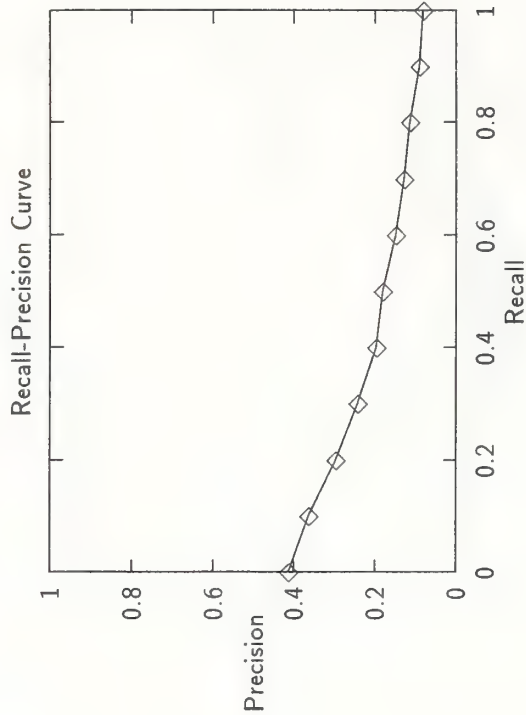
Document Level Averages	
At 5 docs	0.2622
At 10 docs	0.2044
At 15 docs	0.1733
At 20 docs	0.1578
At 30 docs	0.1341
At 100 docs	0.0696
At 200 docs	0.0453
At 500 docs	0.0253
At 1000 docs	0.0153
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2126



Summary Statistics	
Run Number	MTRa961-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	650

Recall Level Precision Averages	
Recall	Precision
0.00	0.4127
0.10	0.3647
0.20	0.2972
0.30	0.2425
0.40	0.1968
0.50	0.1805
0.60	0.1492
0.70	0.1289
0.80	0.1138
0.90	0.0901
1.00	0.0811
Average precision over all relevant docs	
non-interpolated	0.1896

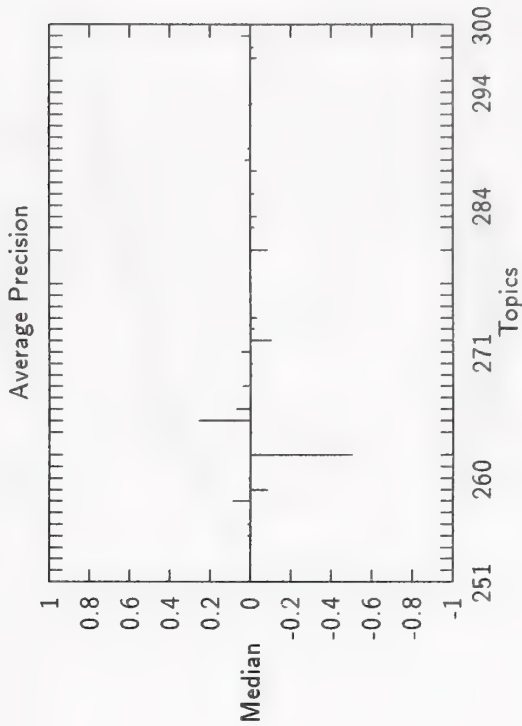
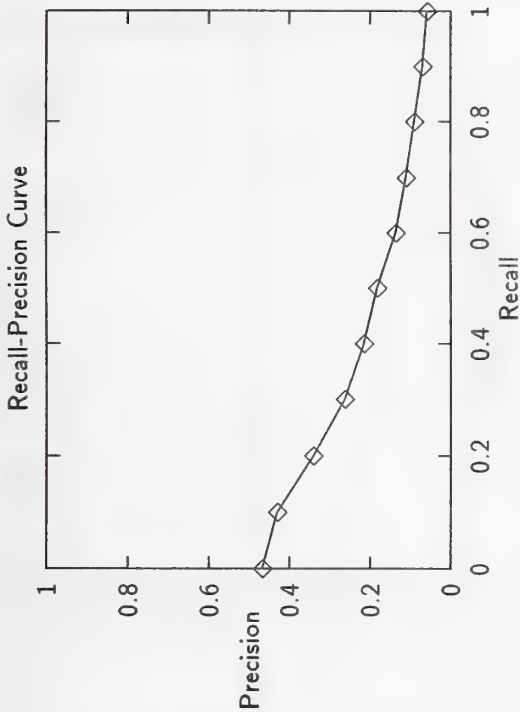
Document Level Averages	
At 5 docs	0.2400
At 10 docs	0.1844
At 15 docs	0.1585
At 20 docs	0.1456
At 30 docs	0.1222
At 100 docs	0.0707
At 200 docs	0.0473
At 500 docs	0.0244
At 1000 docs	0.0144
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1859



Summary Statistics	
Run Number	xerox-nlp1-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
RelRet:	664

Recall Level Precision Averages	
Recall	Precision
0.00	0.4673
0.10	0.4312
0.20	0.3425
0.30	0.2632
0.40	0.2155
0.50	0.1835
0.60	0.1368
0.70	0.1108
0.80	0.0908
0.90	0.0696
1.00	0.0582
Average precision over all relevant docs	
non-interpolated	0.2002

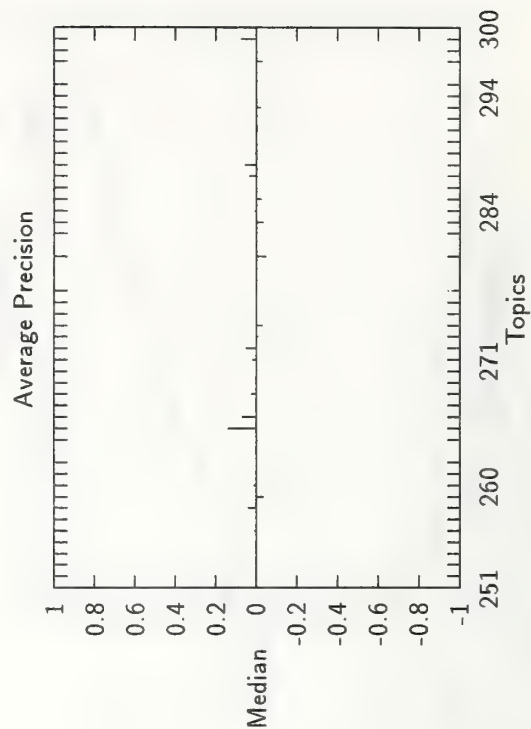
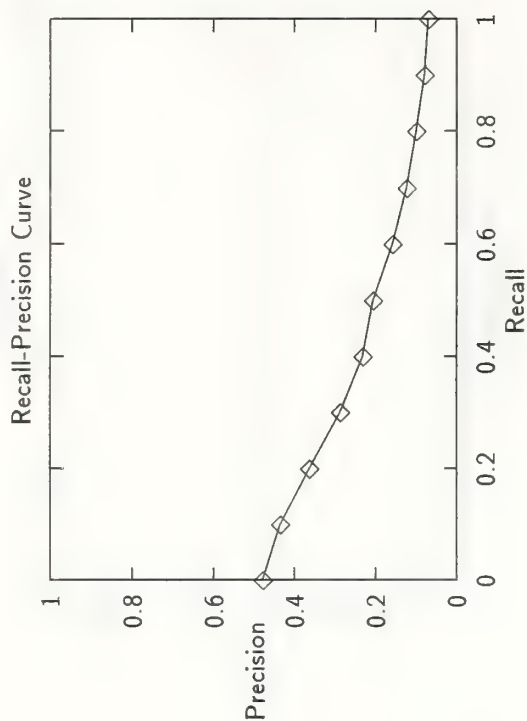
Document Level Averages	
At 5 docs	0.2711
At 10 docs	0.2200
At 15 docs	0.1911
At 20 docs	0.1522
At 30 docs	0.1252
At 100 docs	0.0676
At 200 docs	0.0456
At 500 docs	0.0248
At 1000 docs	0.0148
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2060



Summary Statistics	
Run Number	xerox-nlp2-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	672

Recall Level Precision Averages	
Recall	Precision
0.00	0.4789
0.10	0.4353
0.20	0.3635
0.30	0.2872
0.40	0.2318
0.50	0.2064
0.60	0.1585
0.70	0.1243
0.80	0.1006
0.90	0.0802
1.00	0.0706
Average precision over all relevant docs	
non-interpolated	0.2147

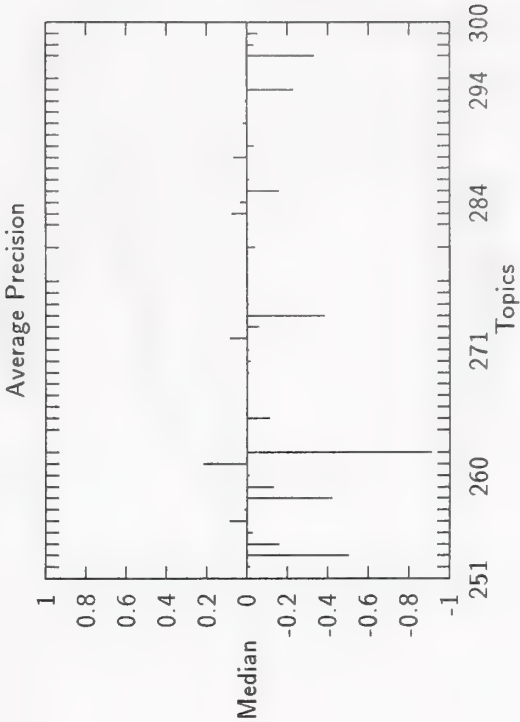
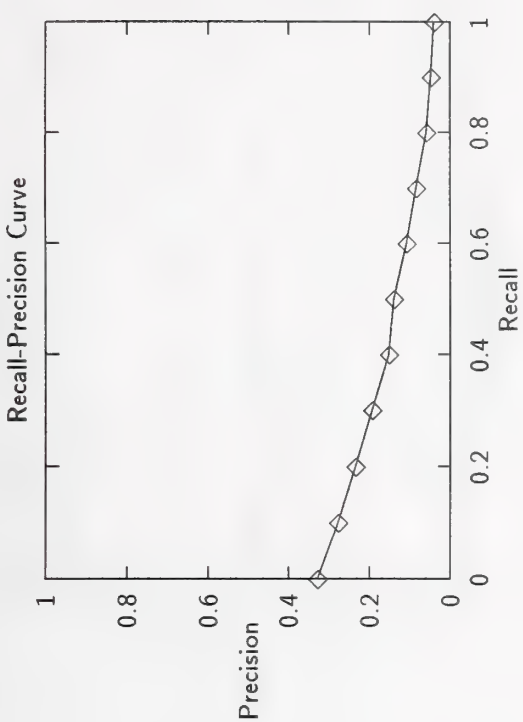
Document Level Averages	
	Precision
At 5 docs	0.2711
At 10 docs	0.2200
At 15 docs	0.1807
At 20 docs	0.1522
At 30 docs	0.1244
At 100 docs	0.0662
At 200 docs	0.0452
At 500 docs	0.0251
At 1000 docs	0.0149
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2172



Summary Statistics	
Run Number	xerox-nlp3-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	613

Recall Level Precision Averages	
Recall	Precision
0.00	0.3292
0.10	0.2780
0.20	0.2355
0.30	0.1943
0.40	0.1526
0.50	0.1394
0.60	0.1089
0.70	0.0840
0.80	0.0591
0.90	0.0469
1.00	0.0396
Average precision over all relevant docs	
non-interpolated	0.1397

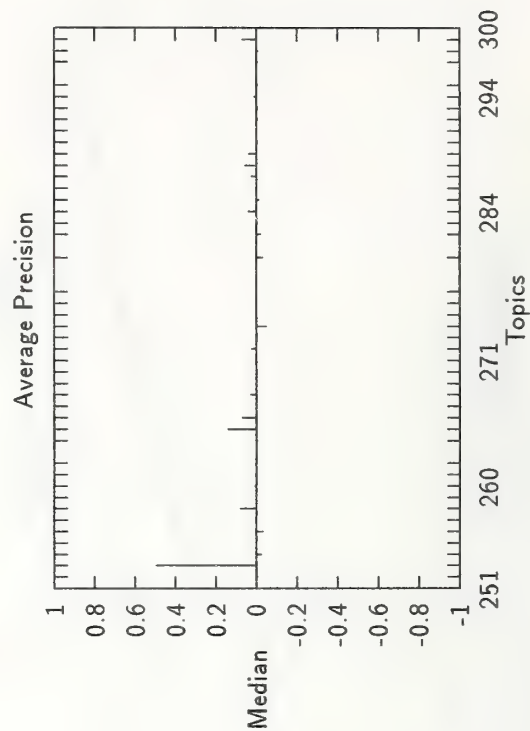
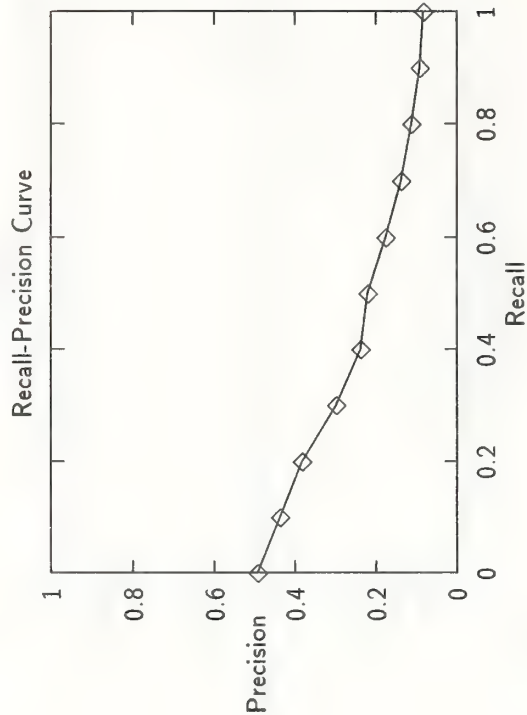
Document Level Averages	
At 5 docs	0.1911
At 10 docs	0.1711
At 15 docs	0.1348
At 20 docs	0.1144
At 30 docs	0.0963
At 100 docs	0.0556
At 200 docs	0.0371
At 500 docs	0.0220
At 1000 docs	0.0136
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1535



Summary Statistics	
Run Number	xerox-nlp4-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	680

Recall Level Precision Averages	
Recall	Precision
0.00	0.4925
0.10	0.4371
0.20	0.3839
0.30	0.2990
0.40	0.2392
0.50	0.2207
0.60	0.1765
0.70	0.1389
0.80	0.1134
0.90	0.0926
1.00	0.0830
Average precision over all relevant docs	
non-interpolated	0.2276

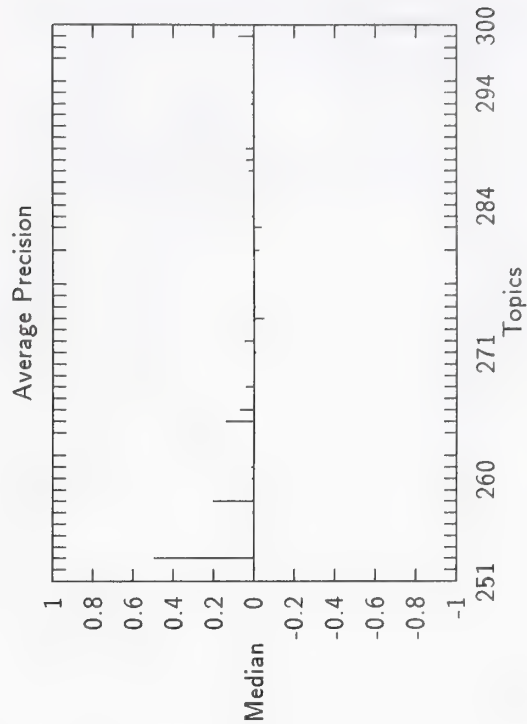
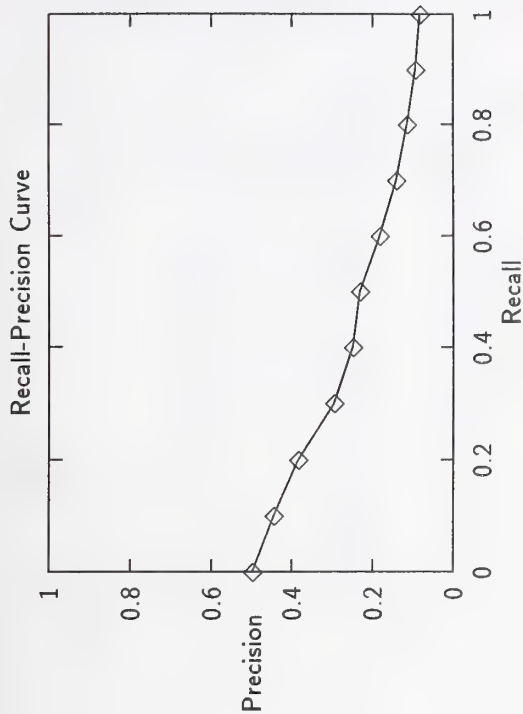
Document Level Averages	
At 5 docs	0.2800
At 10 docs	0.2178
At 15 docs	0.1867
At 20 docs	0.1633
At 30 docs	0.1274
At 100 docs	0.0671
At 200 docs	0.0458
At 500 docs	0.0253
At 1000 docs	0.0151
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2460



Summary Statistics	
Run Number	xerox-nlp5-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
RelRet:	679

Recall Level Precision Averages	
Recall	Precision
0.00	0.4982
0.10	0.4446
0.20	0.3841
0.30	0.2957
0.40	0.2500
0.50	0.2327
0.60	0.1829
0.70	0.1407
0.80	0.1152
0.90	0.0938
1.00	0.0834
Average precision over all relevant docs	
non-interpolated	0.2320

Document Level Averages	
	Precision
At 5 docs	0.2667
At 10 docs	0.2156
At 15 docs	0.1896
At 20 docs	0.1611
At 30 docs	0.1296
At 100 docs	0.0678
At 200 docs	0.0463
At 500 docs	0.0253
At 1000 docs	0.0151
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2488



Summary Statistics	
Run Number	xerox-nlp6-
Number of Topics	45
Total number of documents over all topics	
Retrieved:	45000
Relevant:	1064
Rel_ret:	679

Recall Level Precision Averages	
Recall	Precision
0.00	0.4971
0.10	0.4472
0.20	0.3848
0.30	0.2926
0.40	0.2445
0.50	0.2258
0.60	0.1777
0.70	0.1403
0.80	0.1149
0.90	0.0936
1.00	0.0834
Average precision over all relevant docs	
non-interpolated	0.2295

Document Level Averages	
At 5 docs	0.2756
At 10 docs	0.2178
At 15 docs	0.1852
At 20 docs	0.1633
At 30 docs	0.1296
At 100 docs	0.0676
At 200 docs	0.0463
At 500 docs	0.0252
At 1000 docs	0.0151
R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2473

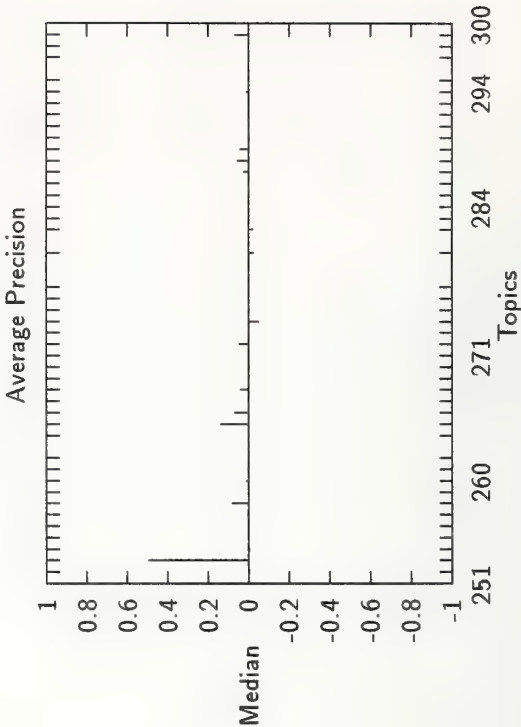
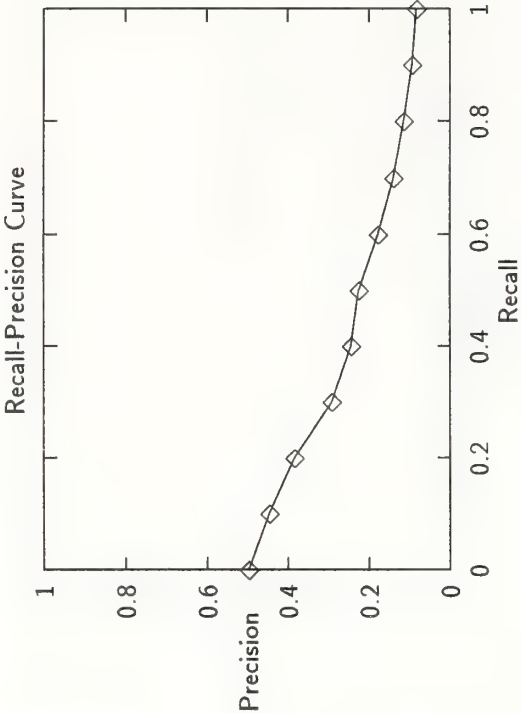
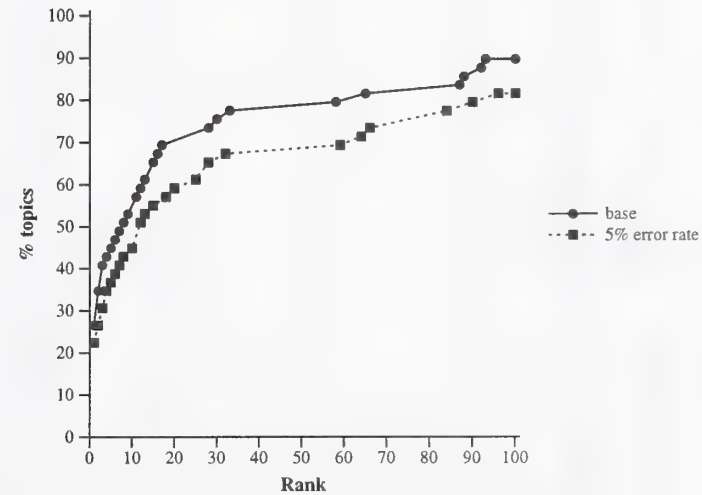


Table 1: Raw Ranks			
	anu5con0	anu5con1	
1	3	7	
2	28	28	
3	5	12	
4	58	96	
5	2000	2000	
6	1	5	
7	1	1	
8	93	84	
9	15	15	
10	1	1	
11	222	351	
12	11	289	
13	87	64	
14	9	12	
15	65	59	
16	6	10	
17	92	207	
18	13	13	
19	11	171	
20	1	1	
21	1	1	
22	2	2	
23	1	1	
24	8	8	
25	15	20	
26	7	66	
27	17	18	
28	2000	2000	
30	3	4	
31	2	84	
32	16	25	
33	1	1	
34	1	1	
35	1	1	
36	1	1	
37	28	28	
38	33	32	
39	221	427	
40	1	1	
41	2	2	
42	4	4	
43	2	6	
44	1	3	
45	1	1	
46	2000	2000	
47	3	3	
48	88	90	
49	12	12	
50	30	201	
Mean rank when found	26.61	53.67	0.00
Mean reciprocal rank	0.3635	0.2992	0.0000

Table 2: Histogram			
Number of items found at rank r where			
	anu5con0	anu5con1	
$1 \leq r \leq 10$	26	22	
$10 < r \leq 100$	18	18	
$100 < r \leq 1000$	2	6	
Not found	3	3	



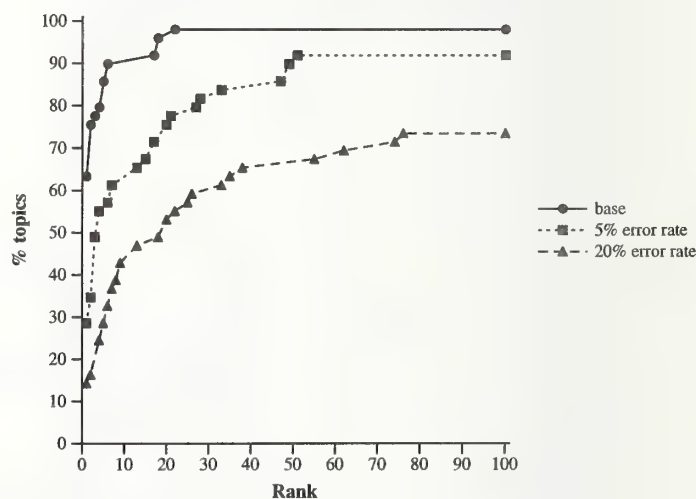
Cumulative % of topics that retrieve target item by given rank

Table 1: Raw Ranks

	CLCONBASE	CLCON5	CLCON20
1	1	3	76
2	1	49	33
3	1	3	7
4	2	3	9
5	5	17	13
6	1	1	1
7	1	1	18
8	1	1	1
9	5	3	1
10	1	4	20
11	1	20	2000
12	1	1	62
13	1	2	1
14	2	2	5
15	1	551	214
16	1	1	4
17	6	47	35
18	1	27	20
19	5	17	2000
20	1	1	927
21	1	1	2
22	1	1	13
23	1	1	4
24	6	7	6
25	2	1	4
26	2	3	1
27	1	49	38
28	18	33	319
30	1	1	26
31	2	7	1
32	1	13	8
33	1	6	25
34	1	4	5
35	1	1	1
36	1	2000	2000
37	18	20	2000
38	4	3	22
39	1	156	2000
40	1	1	55
41	1	15	4
42	1	1	74
43	1	961	897
44	2	2	6
45	1	4	7
46	308	51	268
47	22	13	109
48	3	21	326
49	1	3	9
50	17	28	149
Mean rank when found	9.39	45.02	86.95
Mean reciprocal rank	0.7293	0.4024	0.2138

Table 2: Histogram

Number of items found at rank r where			
	CLCONBASE	CLCON5	CLCON20
$1 \leq r \leq 10$	44	30	21
$10 < r \leq 100$	4	15	15
$100 < r \leq 1000$	1	3	8
Not found	0	1	5



Cumulative % of topics that retrieve target item by given rank

Table 1: Raw Ranks

	CLCONBASE	CLCON5F	CLCON20F
1	1	3	77
2	1	49	27
3	1	3	7
4	2	3	8
5	5	41	13
6	1	1	1
7	1	30	44
8	1	6	3
9	5	3	3
10	1	38	17
11	1	172	2000
12	1	35	122
13	1	6	3
14	2	2	8
15	1	181	272
16	1	3	17
17	6	9	76
18	1	22	8
19	5	17	388
20	1	7	2000
21	1	1	11
22	1	5	1
23	1	9	6
24	6	6	1
25	2	6	4
26	2	14	5
27	1	44	31
28	18	33	431
30	1	17	16
31	2	16	2
32	1	17	3
33	1	7	11
34	1	1	5
35	1	1	1
36	1	2000	2000
37	18	87	2000
38	4	1	6
39	1	103	2000
40	1	83	355
41	1	7	21
42	1	1	93
43	1	74	801
44	2	2	9
45	1	5	5
46	308	28	288
47	22	13	165
48	3	21	518
49	1	13	1
50	17	28	164
Mean rank when found	9.39	26.54	92.00
Mean reciprocal rank	0.7293	0.2297	0.1898

Table 2: Histogram

Number of items found at rank r where			
	CLCONBASE	CLCON5F	CLCON20F
$1 \leq r \leq 10$	44	24	21
$10 < r \leq 100$	4	21	13
$100 < r \leq 1000$	1	3	10
Not found	0	1	5

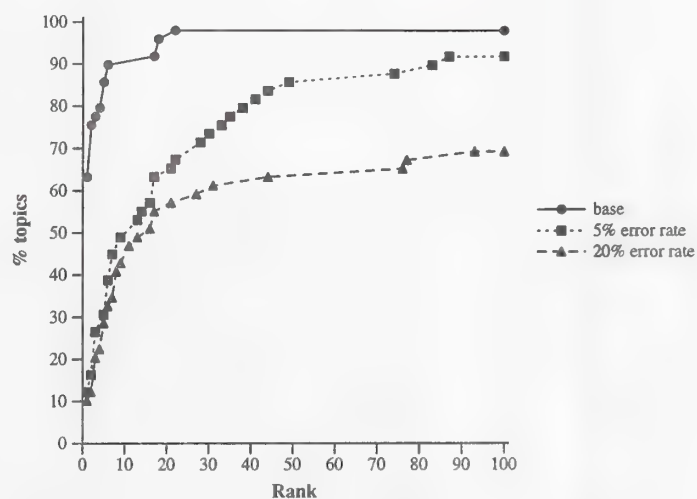
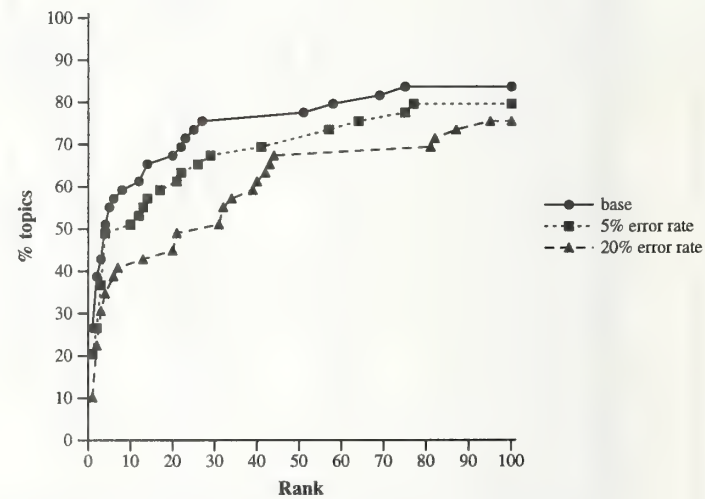


Table 1: Raw Ranks			
	gmu96v01	gmu96v11	gmu96v21
1	3	3	13
2	214	275	136
3	2	3	2
4	69	57	40
5	1	1	1
6	1	1	1
7	75	75	81
8	1	1	1
9	51	57	3
10	3	3	21
11	2	4	2
12	1	978	137
13	23	22	4
14	1	1	3
15	4	282	2000
16	1	1	2
17	1	4	6
18	8	10	20
19	6	21	225
20	1	1	42
21	1	4	3
22	2	2	2
23	2	2	6
24	4	4	2
25	2	2	4
26	4	13	31
27	1	1	3
28	2	3	153
30	58	126	82
31	5	4	2
32	207	170	87
33	12	12	39
34	146	144	146
35	1	1	1
36	261	277	198
37	4	3	32
38	14	17	259
39	20	64	95
40	2000	2000	2000
41	430	327	118
42	5	4	125
43	27	77	34
44	1	1	7
45	1	1	1
46	22	29	21
47	25	26	43
48	475	551	2000
49	14	14	32
50	156	41	44
Mean rank when found	49.38	77.50	50.22
Mean reciprocal rank	0.3856	0.3135	0.2221

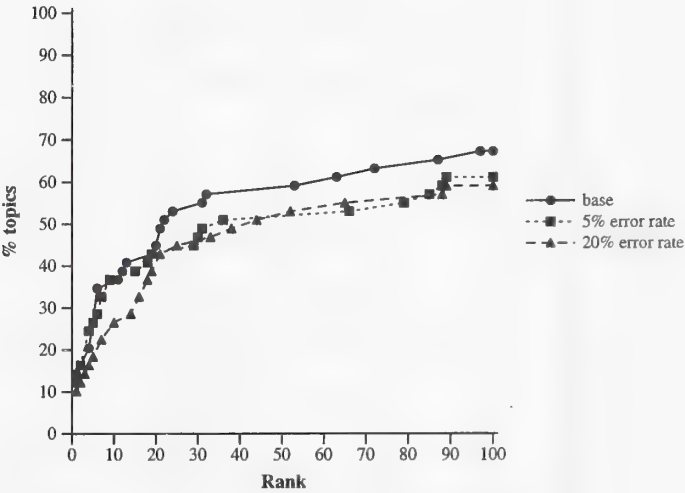
Table 2: Histogram			
Number of items found at rank r where			
	gmu96v01	gmu96v11	gmu96v21
$1 \leq r \leq 10$	29	25	20
$10 < r \leq 100$	12	14	17
$100 < r \leq 1000$	7	9	9
Not found	1	1	3



Cumulative % of topics that retrieve target item by given rank

Table 1: Raw Ranks			
	gmu96v02	gmu96v12	gmu96v22
1	19	19	18
2	851	473	2000
3	6	6	89
4	260	130	188
5	1	1	3
6	20	30	10
7	170	202	343
8	1	1	1
9	87	89	14
10	1	4	195
11	53	66	2
12	97	186	144
13	104	88	1
14	6	9	180
15	72	2000	803
16	266	165	16
17	5	2	10
18	116	243	264
19	24	140	217
20	13	7	217
21	12	36	52
22	6	4	1
23	1	1	4
24	6	4	1
25	5	9	7
26	31	29	44
27	1	1	7
28	21	79	181
30	63	85	65
31	11	15	18
32	361	195	38
33	32	31	25
34	425	981	2000
35	1	1	33
36	2000	2000	932
37	5	4	21
38	22	18	19
39	182	363	310
40	2000	2000	2000
41	2000	240	574
42	4	5	229
43	307	473	135
44	2	2	16
45	1	1	1
46	4	7	5
47	279	277	216
48	975	708	2000
49	21	112	88
50	246	300	21
Mean rank when found	112.96	127.00	127.96
Mean reciprocal rank	0.2039	0.1900	0.1524

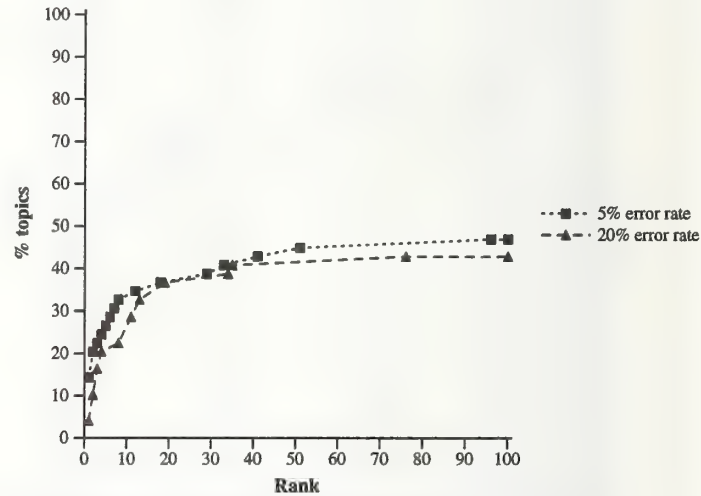
Table 2: Histogram			
Number of items found at rank r where			
	gmu96v02	gmu96v12	gmu96v22
$1 \leq r \leq 10$	17	18	13
$10 < r \leq 100$	16	12	16
$100 < r \leq 1000$	13	16	16
Not found	3	3	4



Cumulative % of topics that retrieve target item by given rank

Table 1: Raw Ranks			
		rutcf2	rutcf1
1		2	2000
2		349	607
3		5	11
4		51	35
5		2000	2000
6		3	11
7		1	2
8		1	1
9		29	19
10		6	3
11		2000	2000
12		2000	2000
13		33	34
14		2000	2000
15		2000	2000
16		2000	2000
17		2000	126
18		2000	2000
19		2000	2000
20		1	3
21		2000	2000
22		4	4
23		1	1
24		8	4
25		18	8
26		2000	2000
27		2	3
28		2000	2000
30		902	779
31		1	13
32		924	941
33		7	19
34		341	638
35		2	2
36		333	271
37		12	11
38		201	641
39		96	509
40		879	849
41		978	972
42		2000	2000
43		405	2000
44		1	13
45		1	2
46		2000	2000
47		41	76
48		2000	2000
49		413	430
50		2000	2000
Mean rank when found	0.00	183.36	219.94
Mean reciprocal rank	0.0000	0.2041	0.1174

Table 2: Histogram		
Number of items found at rank r where		
	rutcf2	rutcf1
$1 \leq r \leq 10$	16	11
$10 < r \leq 100$	7	10
$100 < r \leq 1000$	10	11
Not found	16	17



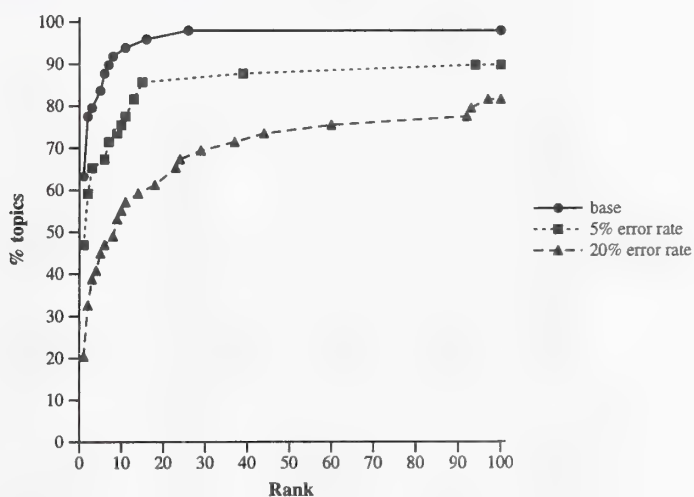
Cumulative % of topics that retrieve target item by given rank

Table 1: Raw Ranks

	ETHFR94	ETHD5N	ETHD20N
1	1	1	2
2	8	15	44
3	2	2	1
4	5	11	24
5	1	1	1
6	1	1	1
7	2	2	2
8	1	1	1
9	2	6	2
10	1	1	8
11	1	1	2
12	1	125	92
13	3	3	3
14	2	1	97
15	1	462	2000
16	1	1	1
17	1	7	11
18	1	2	10
19	1	1	93
20	1	1	9
21	2	2	9
22	1	1	3
23	1	1	2
24	6	7	4
25	1	1	1
26	1	1	2
27	1	1	18
28	6	13	384
30	16	39	60
31	2	2	3
32	7	10	29
33	1	1	1
34	1	3	23
35	1	1	1
36	1	94	981
37	1	1	37
38	1	9	23
39	1	15	342
40	26	138	435
41	1	1	5
42	1	1	5
43	1	13	14
44	1	1	1
45	1	1	1
46	11	103	2000
47	266	119	186
48	2	3	425
49	1	1	6
50	5	2	156
Mean rank when found	8.24	25.10	75.77
Mean reciprocal rank	0.7353	0.5737	0.3218

Table 2: Histogram

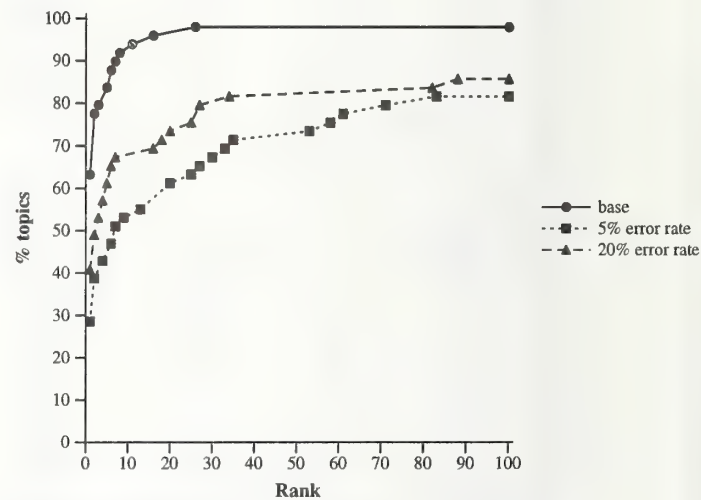
Number of items found at rank r where			
	ETHFR94	ETHD5N	ETHD20N
$1 \leq r \leq 10$	45	37	27
$10 < r \leq 100$	3	7	13
$100 < r \leq 1000$	1	5	7
Not found	0	0	2



Cumulative % of topics that retrieve target item by given rank

Table 1: Raw Ranks			
	ETHFR94	ETHD5P	ETHD20P
1	1	1	1
2	8	20	6
3	2	13	2
4	5	1	3
5	1	1	1
6	1	1	1
7	2	27	1
8	1	1	1
9	2	1	6
10	1	2	1
11	1	1	1
12	1	25	400
13	3	30	3
14	2	4	2
15	1	2000	933
16	1	1	1
17	1	20	185
18	1	9	1
19	1	199	641
20	1	2	1
21	2	7	2
22	1	4	1
23	1	6	1
24	6	2	5
25	1	1	1
26	1	1	1
27	1	33	1
28	6	490	7
30	16	133	2000
31	2	1	2
32	7	53	4
33	1	2	1
34	1	1	1
35	1	2	4
36	1	7	82
37	1	20	16
38	1	58	1
39	1	35	27
40	26	2000	88
41	1	83	25
42	1	192	20
43	1	71	5
44	1	1	1
45	1	1	1
46	11	2000	443
47	266	418	645
48	2	6	18
49	1	61	34
50	5	2000	27
Mean rank when found	8.24	45.51	76.15
Mean reciprocal rank	0.7353	0.3720	0.4978

Table 2: Histogram			
Number of items found at rank r where			
	ETHFR94	ETHD5P	ETHD20P
$1 \leq r \leq 10$	45	26	33
$10 < r \leq 100$	3	14	9
$100 < r \leq 1000$	1	5	6
Not found	0	4	1



Cumulative % of topics that retrieve target item by given rank

APPENDIX B: SUMMARY PERFORMANCE COMPARISONS TREC-2, TREC-3, TREC-4, TREC-5

Karen Sparck Jones
Computer Laboratory, University of Cambridge

February 10, 1997

These comparisons are designed to allow trends over successive TRECs to emerge. But they are selective views, and do not summarise all there is to say about TREC overall. (TREC-1 is not included as it was a start-up, debugging, cycle.)

The performance comparisons between TREC-2, TREC-3 and TREC-4 originally appeared as Appendix C to the TREC-4 Proceedings, ie to 'The Fourth Text REtrieval Conference (TREC-4)' Ed D.K. Harman, NIST Special Publication 500-236, National Institute of Standards and Technology, 1996 (and see also the opening Overview paper by Donna Harman). The TREC-5 figures are from the TREC-5 conference working papers.

The tables cover Adhoc and Routing task results respectively. The former was required, the latter recommended, up to TREC-4. Adhoc runs were still required for TREC-5; and the TREC-5 Routing results are also given for completeness. However Interactive, as a separately defined task, though covered in the previous comparisons for TREC-3 and TREC-4, is not included here as the number of participants has remained small. Other TREC tracks are not covered either.

In the previous comparisons, changes in the nature of the topics used in successive TRECs were disregarded, and no distinctions were made between automatic and manual search queries. However for the Adhoc case in particular, further changes in the character of the supplied topic data, and in the precise specification of the options within the Adhoc task, make simple continuity of comparison through TREC-5 more problematic.

Thus for Adhoc up to TREC-4, the quantity and quality of content supplied per topic was reduced. But automatic and manual query formation were accepted as legitimate alternatives. So in the previous comparison tables the best performing of the official runs submitted per team was used, regardless of whether this was obtained by automatic or manual processing. However in TREC-5, the Adhoc topics were 'split' to give two forms. Short (S) and Long (L), with the latter adding content to the former. S was obligatory for automatic searching. L was supplied for manual searching, which could also be rather more interactive than in previous TRECs, and was also optional for automatic.

The data and relevant run sets used for the main comparison tables below are therefore as follows:

Adhoc

	TREC-1	TREC-2	TREC-3	TREC-4	TREC-5	
topic components					S	L
title	x	x	x			x
description	x	x	x	x	x	x
narrative	x	x	x			x
concepts	x	x				
queries	a/m	a/m	a/m	a/m	a	m/(a)

Routing

topics for TREC-1 – 3 were in the same style as Adhoc TREC-1 – 2
 topics for TREC-4 – 5 were subsets drawn from all previous topics,
 so had variable composition.

Tables 1 and 2 present Precision performance at Document Cutoff 30. The data are only for full Category A runs, not Category B, and cover only the higher levels of performance, not all the submitted runs.

The conventions are as follows: figures are not rounded; performance is assigned to ‘blocks’; teams per block are NOT in merit order, but in Workshop Notes or Proceedings order; the best of two official runs is taken, regardless of the particular strategy used, where there are two and these are deemed legitimate alternatives.

COMMENTS:

For both Adhoc and Routing, constant participants who initially performed well have continued to do so, though they have sometimes not profited from experiments, while others who started less well have successfully improved their performance. However many teams have not participated throughout the series, or have more recently concentrated on the tracks, so no inferences should be drawn where teams figure only occasionally in the tables.

Adhoc

1. Ad hoc best performance improved from TREC-2 to TREC-3, even though the TREC-3 topics were less rich. The sharp fall in performance for TREC-4 must reflect the minimal topics given for the tests. The further decline in TREC-5, even for the L topics which were fuller than the TREC-4 topics and like the TREC-3 ones, reflects the fact that the topics were deemed ‘difficult’ in relation to the definition of relevant documents.
2. The generally low levels of performance (even for the better-performing teams) in TREC-4 and TREC-5 must be taken as representing a more realistic retrieval situation than TREC-2 and TREC-3, as far as the S topics are concerned, though how far the TREC-5 topic ‘difficulty’ is representative is not clear.
3. In TREC-2 and TREC-3 automatic query formation was more common than manual, and often performed well, appearing even in the top blocks. Indeed there was relatively more use of automatic query in TREC-3 than TREC-2. But in TREC-4 there was a clear shift towards manual, doubtless in response to the perceived need to beef up the initial minimal

Precision	TREC-2	TREC-3	TREC-4	TREC-5		
	a/m	a/m	a/m	Short a	Long m	Long a
≥ 60		UMass City Berkeley				
≥ 55	UMass HNC VT	Cornell Mead				
≥ 50	Cornell Berkeley Dortmund CMU/Clarit Verity Siemens CUNY	VT Westlaw ETH CUNY				
≥ 45	City Bellcore ETH RMIT Conquest	NYU CMU/Clarit RMIT RutgersK	Excalibur/ Conquest CUNY Waterloo		ETH	
≥ 40	Berkeley Clarit/CMU Cornell GMU UMass InText ANU		Waterloo	
≥ 35	City GE/NYU	**Lexis	ANU Clarit Cornell GE/NYU GMU Lexis	
≥ 30		OpenText CUNY Berkeley	City CUNY ETH
≥ 25	Apple City Cornell IBMTJW ETH UMass	DCU IBM	Apple GE/NYU RMIT Berkeley

Table 1: Average precision after 30 documents retrieved for the Adhoc task

Precision	TREC-2	TREC-3	TREC-4	TREC-5
≥ 60	Cornell Dortmund	City		
≥ 55	City Berkeley UMass Bellcore CMU/Clarit CUNY	UMass Cornell Berkeley Dortmund Bellcore	City UMass Xerox	
≥ 50	Rutgers HNC GE TRW Verity Siemens	CMU/Clarit Westlaw Logicon TRW Florida	Cornell CUNY	City Cornell UMass
≥ 45	VT	Xerox NYU Verity ETH NSA NEC	Logicon GE/NYU	CUNY
≥ 40	GE/NYU ETH Berkeley

Table 2: Average precision after 30 documents retrieved for the Routing task

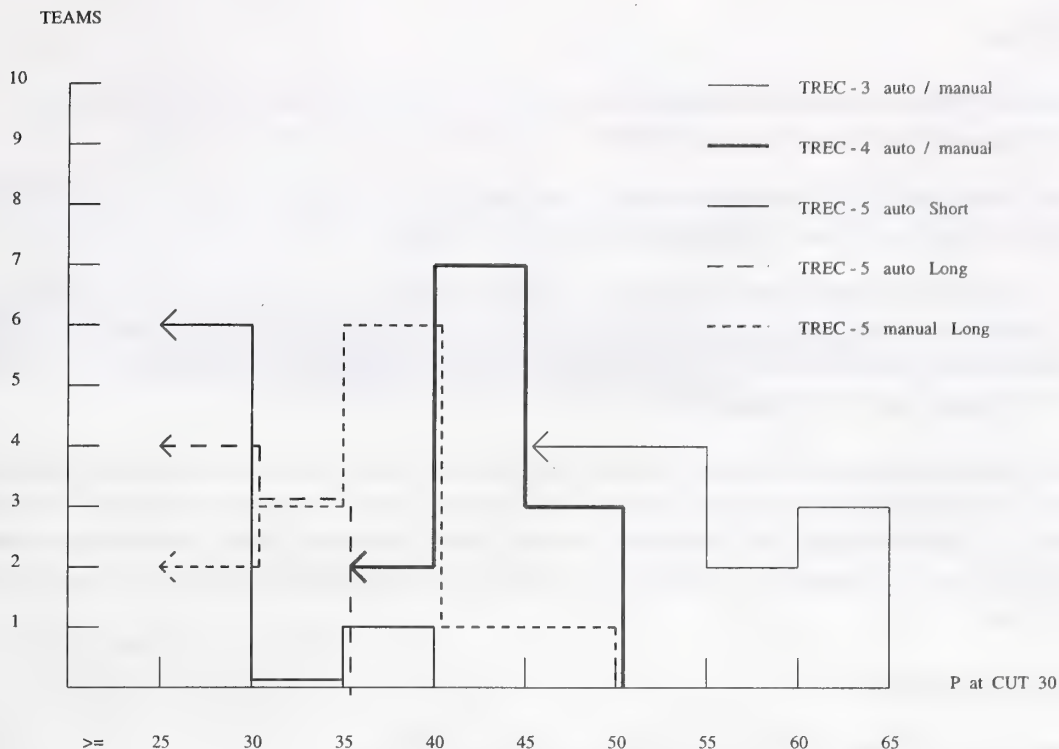


Figure 1: Teams performance distribution (upper levels) TREC-3 – 5 for the Adhoc task

topics, with almost all the teams covered by the table using manual queries. However at least one of the top-level teams using automatic query (Cornell) continued to do comparatively well in TREC-4. It is evident that manual query formation was advantageous for TREC-5 even when the same initial topic information (L) was used. But the L topics supplied more information for query formation than the S ones, so the comparison between S automatic queries and L manual ones for TREC-5 is not strictly fair. It is not completely evident what inferences are to be drawn from the outlying lead results in TREC-5. (Late Note: After the TREC-5 conference, the Lexis result marked ** in Table 1 was reclassified from automatic to manual.)

The histogram in Figure 1 shows the table data for TREC-3 – 5 in a more direct form (TREC-2 is omitted because of the ‘over-good’ topics.) The bars in the histogram show the number of teams that achieved a precision value (for Document Cutoff 30) in each given range. The backward-pointing arrows simply indicate that there were other teams with lower performance that are not included in any of the analysis in this Appendix.

It is important to note that the definition “manual” covers a wide range of human effort from the fairly minimal to the very intensive, and was also explicitly widened in TREC-5 to allow ‘feedback’ strategies. It is nevertheless not clear, in general, what forms of manual effort are especially profitable, or how far intensive effort (and hence time) pays off, or how manual input and automatic devices are best combined. In earlier TRECs it appeared that relatively modest effort could deliver as well as much more intensive work. Detailed analysis is needed for TREC-4 and TREC-5.

Routing

1. Routing has shown a slow decline in overall performance, again reflecting less good topic starting information: thus both TREC-4 and TREC-5 had 'tough' topics.
2. But when topics are less problematic, and there is rich training data (as with TREC-2 and TREC-3) a good level of performance can be obtained.
3. In TREC-2 automatic query formation was only slightly more common than manual, but by TREC-5 manual formation (for the teams shown in the table) had disappeared, reflecting the value of the large training data availability and its utility in compensating for any weakness in automatic query management.

Comparing Adhoc and Routing results, performance in TREC-2 and TREC-3 reached similar levels, attributable to the good topic specifications for the former even though Routing also benefitted in the same way and from the training data. However Adhoc performance has fallen much below that for Routing in TREC-4 and TREC-5, since Routing has been able to benefit from training data.

OVERALL REMARKS:

1. Many (very) different approaches give similar performance.
2. The overall results, and the general findings, for early TRECs reported in 'Reflections on TREC', Information Processing and Management 31 (3), 1995, p 309 and p 311 respectively, still essentially hold.
3. However while for earlier TRECs some convergence could be detected among practitioners of the statistical approach, the changing data conditions mean that strategy details are open to investigation.
4. More generally the range of devices, and of combinations of devices, in TREC remains very wide. So while one natural conclusion is that relatively simple and wholly or largely automated approaches can perform reasonably, the effect of the data changes over successive TRECs show that more understanding of the effects of environment variables on system parameters, for large text files, is required. As already noted, a detailed comparative analysis of what manual query formation involves and achieves, and how it related to independent rather than query-dependent document indexing, would be very useful.

NB: as in the previous comparisons, all the points made here are broad brush ones; and the nature of the tables must always be borne in mind. Thus there may be real differences within performance blocks, and also none between members of adjoining blocks. However as it is not obvious what performance differences are statistically significant, what differences are meaningful to users, and how significance and meaning are related, it is only proper to take a generally rather conservative view of apparent performance differences in the tables. More concretely, Precision of 55% and 45% are respectively equivalent to 16.5 and 13.5 relevant documents retrieved, a difference which may not matter much to a user.

ANNOUNCEMENT OF NEW PUBLICATIONS ON INFORMATION TECHNOLOGY

Superintendent of Documents
Government Printing Office
Washington, DC 20402

Dear Sir:

Please add my name to the announcement list of new publications to be issued in the series: National Institute of Standards and Technology Special Publication 500-.

Name _____

Company _____

Address _____

City _____ State _____ Zip Code _____

(Notification key N-503)

NIST Technical Publications

Periodical

Journal of Research of the National Institute of Standards and Technology—Reports NIST research and development in those disciplines of the physical and engineering sciences in which the Institute is active. These include physics, chemistry, engineering, mathematics, and computer sciences. Papers cover a broad range of subjects, with major emphasis on measurement methodology and the basic technology underlying standardization. Also included from time to time are survey articles on topics closely related to the Institute's technical and scientific programs. Issued six times a year.

Nonperiodicals

Monographs—Major contributions to the technical literature on various subjects related to the Institute's scientific and technical activities.

Handbooks—Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications—Include proceedings of conferences sponsored by NIST, NIST annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, and bibliographies.

National Standard Reference Data Series—Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a worldwide program coordinated by NIST under the authority of the National Standard Data Act (Public Law 90-396). NOTE: The Journal of Physical and Chemical Reference Data (JPCRD) is published bimonthly for NIST by the American Chemical Society (ACS) and the American Institute of Physics (AIP). Subscriptions, reprints, and supplements are available from ACS, 1155 Sixteenth St., NW, Washington, DC 20056.

Building Science Series—Disseminates technical information developed at the Institute on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.

Technical Notes—Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NIST under the sponsorship of other government agencies.

Voluntary Product Standards—Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The standards establish nationally recognized requirements for products, and provide all concerned interests with a basis for common understanding of the characteristics of the products. NIST administers this program in support of the efforts of private-sector standardizing organizations.

Order the following NIST publications—FIPS and NISTIRs—from the National Technical Information Service, Springfield, VA 22161.

Federal Information Processing Standards Publications (FIPS PUB)—Publications in this series collectively constitute the Federal Information Processing Standards Register. The Register serves as the official source of information in the Federal Government regarding standards issued by NIST pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations).

NIST Interagency Reports (NISTIR)—A special series of interim or final reports on work performed by NIST for outside sponsors (both government and nongovernment). In general, initial distribution is handled by the sponsor; public distribution is by the National Technical Information Service, Springfield, VA 22161, in paper copy or microfiche form.

U.S. Department of Commerce
National Institute of Standards
and Technology
Gaithersburg, MD 20899-0001

Official Business
Penalty for Private Use \$300